



UNIVERSIDAD DE  
SAN BUENAVENTURA  
SEDE BOGOTÁ

# SOFTWARE ENGINEERING:

METHODS, MODELING,  
AND TEACHING

VOLUME # 4

## COMPILATORS

Carlos Mario Zapata Jaramillo  
Claudia Elena Durango Vanegas  
Wilder Perdomo Charry



# SOFTWARE ENGINEERING:

METHODS, MODELING,  
AND TEACHING

VOLUME # 4

**Software engineering : methods, modeling, and teaching, Volume # 4/** Carlos Mario Zapata Jaramillo, Claudia Elena Durango Vanegas, Wilder Perdomo Charry, compiladores. – Bogotá : Editorial Bonaventuriana, 2017.

465 páginas.; Ilustraciones a color.

Incluye referencias bibliográficas.

ISBN: 978-958-8928-49-4

1. Ingeniería de software. – 2. Análisis de sistemas. – 3. Programación (Computadores electrónicos). – 4. Tecnología educativa. – I. Zapata Jaramillo, Carlos Mario. – II. Durango Vanegas, Claudia Elena – III. Perdomo Charry, Wilder.

**CDD. 005.1**



**Software Engineering:  
Methods, Modeling, and Teaching, Volume # 4**

© Universidad de San Buenaventura  
© Universidad Nacional de Colombia

© Editorial Bonaventuriana, 2017  
Universidad de San Buenaventura, sede Bogotá  
Carrera 8 H N.º 172-20,  
PBX: 57 (1) 667 1090  
[www.usbbog.edu.co](http://www.usbbog.edu.co)  
Bogotá - Colombia

Rector: Fray José Wilson Téllez Casas, O.F.M.  
Coordinador editorial: Pablo Enrique Sanches Ramírez  
Jefe Unidad de Comunicaciones y Protocolo: Luis Alfredo Téllez Casas  
Diseño y diagramación: Luis Orlando Ferrucho Branz

**Aviso Legal**

Los editores y los autores son responsables del contenido de la presente obra

Prohibida la reproducción total o parcial de este libro por cualquier medio, sin permiso escrito de la Editorial Bonaventuriana

Derechos reservados de la Universidad de San Buenaventura

**ISBN:** 978-958-8928-49-4

**Depósito legal:** se da cumplimiento a lo estipulado en la Ley 44 de 1993, Decreto 460 de 1995 y Decreto 358 de 2000.

Impreso en Colombia - *Printed in Colombia.*

# SOFTWARE ENGINEERING:

---

## METHODS, MODELING, AND TEACHING

---

### VOLUME # 4

#### COMPILATORS:

**Carlos Mario Zapata Jaramillo**

Profesor Titular

Universidad Nacional de Colombia Sede Medellín

correo: cmzapata@unal.edu.co

**Claudia Elena Durango Vanegas**

Profesora Investigadora

Universidad de San Buenaventura, seccional Medellín

correo: claudia.durango@usbmed.edu.co

**Wilder Perdomo Charry**

Profesor Investigador

Universidad de San Buenaventura, seccional Medellín

correo: wilder.perdomo@usbmed.edu.co



**UNIVERSIDAD DE  
SAN BUENAVENTURA**  
SEDE BOGOTÁ



**UNIVERSIDAD  
NACIONAL  
DE COLOMBIA**  
SEDE MEDELLÍN





# Table of Content

---

<b>PREFACE</b> .....	8
----------------------	---

## **PART 1 – METHODS**

<b>Chapter # 1</b>	
Template for describing patterns of interaction and user experience.....	11
<b>Chapter # 2</b>	
Using ISO/IEC 29110 Deployment Package to construct educational video games in software engineering .....	27
<b>Chapter # 3</b>	
Associating quality measures to the alpha states of the SEMAT kernel.....	41
<b>Chapter # 4</b>	
Computational tool for a communication system for persons with tetraplegia using an eye-tracking sensor.....	55
<b>Chapter # 5</b>	
Driving Security Aware Android Application Development Based on Malware Analysis Data Visualization .....	77
<b>Chapter # 6</b>	
A Methodology to Assist Novice Engineers to Produce Quality Research and Development Projects .....	98
<b>Chapter # 7</b>	
Improving Privacy Notices Usability Applying Cognitive Ergonomics in Interaction Patterns .....	122
<b>Chapter # 8</b>	
Analysis methodology for quality source code in software development.....	143
<b>Chapter # 9</b>	
The importance of functional and data requirements in supporting the adoption process of EHRS .....	161
<b>Chapter # 10</b>	
Towards an Enterprise Architecture framework for an IT SME: Miracle Business Network .....	175

## **Chapter # 11**

User-oriented application for source code metrics definition and extraction based on a metrics framework.....	193
--	-----

## **Chapter # 12**

PMBOK and Essence: Partners for IoT Projects .....	211
--	-----

## **PART 2 - MODELING**

## **Chapter # 13**

Automotive Safety Requirements Specification .....	225
--	-----

## **Chapter # 14**

Conceptual synthesis of practice as a theoretical construct in Software Engineering .....	244
--	-----

## **Chapter # 15**

Facilitating the development of Collaborative Applications with the MVC Architectural Pattern .....	268
--	-----

## **Chapter # 16**

Creating an Estimation Model from Functional Size Approximation Using the EPCU Approximation Approach for COSMIC (ISO 19761) .....	291
--	-----

## **Chapter # 17**

A Representation Based in SEMAT Kernel of the Test Planning Process According to ISO/IEC/IEEE 29119-2 Standard .....	311
---	-----

## **Chapter # 18**

A KAOS representation by using the SEMAT kernel .....	322
---	-----

## **Chapter # 19**

QUACOP: An approach to Increase the Quality of Artifacts considered in a Project Planning Process.....	338
---	-----

## **Chapter # 20**

New relationships of the Risk alpha with the Semat Essence kernel.....	352
--	-----

## **PART 3 - TEACHING**

## **Chapter # 21**

Towards a Compilation of Problems in the Adoption of Agile-Scrum Methodologies: A Systematic Literature Review.....	364
--	-----

**Chapter # 22**

An Instructional Proposal for study of concepts on  
Software Engineering assisted by Ludic Virtual Learning Environments ..... 390

**Chapter # 23**

Augmented Reality Applied in the Museum of Memory of Tlaxcala .....410

**Chapter # 24**

Promoting Software Engineering Concepts  
to Children through a Serious Game .....422

**Chapter # 25**

Representation of teaching and learning practices  
about embedded systems using a SEMAT kernel extension.....443





# Preface

---

Software Engineering in Latin America is growing stronger. Too much work has been devoted to this discipline and elsewhere we have evidence about this fact. Methods, modeling, and teaching are ways to classify the work Latin American researchers are doing: the fourth volume of this book is now full of the effort made in our discipline. Since the first volume of this book, we are promoting new ideas coming from researchers about software engineering. In this case, we have our three common knowledge areas. The names of the authors are probably different, but the intention is the same: promoting the research and practice of software engineering in Latin America.

This book should be read by all audiences interested in software engineering, from students to real-world practitioners. Each chapter is self-contained, but—as a reader—you need a minimum background about each topic. We strongly recommend you to read the introduction of each chapter before starting to go deep in the contents, just to be sure you have enough background about the topic you are reading. When you feel sure about it, you can proceed to the detailed information of the chapter.

We organized the chapters in the same way we do in the previous three volumes: Part one is devoted to methods, Part two is devoted to modeling, and Part three is devoted to teaching.

The first part—methods—includes studies about: usability and accessibility made by using interaction patterns; international standards like ISO, SEMAT, and PMBOK; malware detectors; automated research assistants; source code inspections by using metrics frameworks; electronic health record systems; and enterprise architecture frameworks.

The second part—modeling—comprises studies about: safety requirements specifications; practices and risk alpha as theoretical constructs; MVC patterns; functional size estimation; test planning and KAOS based on the SEMAT kernel; and data quality.

The third part—teaching—is related to: agile methods; software engineering concepts by using serious games and virtual learning environments; augmented reality; and embedded systems by using the SEMAT kernel.

As you can see, we have full coverage of software engineering topics, so you can have a landscape vision about the software engineering research in Latin America or you can go deeper in the topic you select. Whatever reason you have for reading this book, we promise you will be rewarded for your reading.

**Carlos M. Zapata, Claudia E. Durango, Wilder Perdomo**  
**Compilers**



# **PART 1**

# **METHODS**

---

# Chapter # 1

## Template for describing patterns of interaction and user experience

Yuliana Puerta Cruz  
Universidad del Cauca  
Popayán-Colombia  
Fundación Universitaria  
Tecnológico Comfenalco  
Cartagena-Colombia  
puertacruz@gmail.com

Cesar A. Collazos  
Universidad del Cauca  
Popayán -Colombia  
ccollazo@unicauca.edu.co

Josefina Guerrero García,  
Juan González Calleros  
Benemerita Universidad  
de Puebla  
Puebla- México  
joseguga01@gmail.com

### 1. Introduction

Interaction patterns represent a solution to various problems of interaction design, especially those looking to accelerate the development, allowing implement effective solutions to common problems by avoiding the need to evaluate and re-evaluate every aspect of a project [1].

These patterns [2] are directly related to the representation of information according to user needs, and this refers to the efficiency and user satisfaction in their experience with software products.

A pattern language is defined as the specification of a number of elements (patterns) and their relationships (with other patterns) so that it managed to describe good solutions to the various problems that arise in a specific [3] context.

This work was performed as part of the development of a research project that aims to define a pattern language interaction special focus on the user experience. The main objective is to define a template for describing patterns of interaction that will be designed.

In the definition of the template describing patterns, it has conducted a review of the structure definition of patterns of interaction proposed by different authors, in order to analyze the proposed features in these templates to identify which of these features to



consider when designing a pattern language interaction. In addition, it has designed and implemented a survey instrument among the community of academic and professional experts in the field of interaction patterns to corroborate the findings of conceptual analysis.

They have also reviewed aspects that define the user experience (UX), for which have been considered proposals of different authors such as: [4], the Panel Morville [5] and mainly review in [6]. These facets have been revised considering that the main objective of this work focuses on ensuring the scope of software products with good levels of user experience.

This chapter is organized in the following structure: section 2 shows a state of the art review of the concept of interaction patterns and languages, as well as the main structures for describing patterns of interaction. Section 3 the instrument used to collect information in the expert community patterns and an analysis of the survey results, section 4 presents the template selected and the criteria under which the selection was described. Finally, conclusions and future work.

## 2. Patterns of interaction

Patterns of interaction [7] are also known as patterns HCI or patterns design user interfaces, in this context it spoke of patterns of interaction, these have to do with the representation of information according to user needs, and results in satisfaction, efficiency and acceptability as perceived by users to use the software products they require.

According to [8] Alexander's original ideas it has been possible to move the process of interaction, thus interaction patterns help to design user-friendly systems for people.

UI design patterns are solutions that solve common design problems recurring. Design patterns are the standard benchmarks for user interface designer.

A pattern language is a network of closely intertwined patterns defining a process to systematically solve a series of problems related and interdependent development of software [9].

In [7] the patterns of interaction are defined as a proven solution for professional interaction design, usability and user experience that provides best practices for designing human interaction computer to any of the phases the design, engineering, evaluation or use of interactive systems, generally characterized by the user interface.

Pattern languages are structured to describe good design practices containing a collection of interrelated standards that aim to disseminate the body of knowledge contained method. Describe the key features of effective solutions to fulfill various design objectives [10].

In [11] a pattern language is defined as “The specification of a number of elements (patterns) and their relationships (with other patterns) so that allow us to describe good solutions to the various problems that arise in a specific context.

Pattern languages have three essential elements or activities, such as the standard definition patterns, grouping patterns and description of relationships between patterns. The standard definition refers to the definition of a structure describing patterns, necessary to characterize patterns. Grouping patterns must be made from a categorization that allows organize them according to certain characteristics defined. While relationships should be described using some type of mapping or diagram [7].

Considering that in the future, since this proposal will develop a language of patterns of interaction –oriented user experience, is assumed to define a structure for describing patterns.

## 2.1 Describing Patterns of Interaction

Interaction patterns must have a structure design, the state of the art reveals various proposals for these structures, which should facilitate communication between designers, and almost all contain an important set of content suggested by Gamma [11]. Here we review some of the structures used by various authors. Table 1 shows the elements proposed by Jennifer Tidwell [12].

**Table 1. Description of Patterns Tidwell [12]**

Name:	Name by which the pattern is identified.
Problem:	This item describes the situation that the pattern will solve.
Context:	User features and characteristics of the tasks to be performed.
Forces:	How they influence different aspects of the problem.
Solution:	Clear description of the proposed solution.
Consequences	Describes the results of applying the pattern.
Usability principles:	Describes the principles or ergonomic criteria on which the employer is based.
Examples:	An illustrative example of a successful solution.

Another important proposal of pattern languages is that of Van Welie [8, 1], items that are included in his proposal are listed below in Table 2.

**Table 2. Description of patterns van welie [1]**

Name:	Pattern title, which must be representative, clear and concise concept you want to communicate.
Context:	A description of the situation in which the pattern can be used, what are the characteristics of the context, in terms of tasks, the user.
Forces:	Contextual aspects that need to be optimized.
Solution:	Clear description of the proposed solution.
Consequences	Describes the results of applying the pattern.
Examples:	An illustrative example of a successful solution.

Table 3 describes the proposal of Vanderdonckt [13] for describing patterns of interaction.

**Table 3. Description of Patterns Vanderdonckt [13]**

Problem	It describes the situation that the pattern will solve.
Context	Characteristics of the context in which the problem occurs are described
Forces	Aspects that influence with great importance in the situation.
Solution	Description of the proposed solution.
Comments	Additional information that enables implementation of pattern

Van Duyne [14] proposes the use of six key features or elements which are described in Table 4.

**Table 4. Description of patterns Van Duyne [14]**

Title Pattern	It refers to the name of the pattern.
Background	Context pattern describes the relationship of this with other patterns.
Forces	Describes in more detail people, tasks, technology and society affect design problems.
Solution:	It shows how to solve the problem, provide an outline of how to solve the problem
Consequence	Describes the results of applying the pattern.
Other Patterns	Other patterns that help to complete this pattern are recommended

The template description of the Master Detail Patterns [15] is described in Table 5.

**Table 5. Description of Patterns Master Detail [15]**

Pattern Name:	It refers to how the pattern will be appointed.
Also known as:	Another name for the pattern.
Classification:	It represents the type of pattern:
Motivation or Problem:	What is the sample scenario to implement this pattern?
Solution:	What problems are solved patterns?
Constraint:	What restrictions are required?
Forces:	Advantages and strengths of using
Weakness:	Disadvantages or limitations to use this patterns.
Justification:	What is the story behind this pattern, because it works?
Applicability or content:	When this pattern is applied?
Context of use:	What are the category of user, the environment and the platform that this pattern can apply?
Structure	What are the class hierarchy diagrams for objects in this pattern?
Competitor:	What are the objects participating in this pattern?
Consequences:	What are the advantages and disadvantages of using this pattern?
Implementation:	What techniques or problems arise in the application of these patterns?
Know uses:	What are some examples of real systems using this pattern?
Related Patterns:	What other models in this collection are related to pattern this pattern?

Table 6 shows a format for documenting patterns in order to minimize communication problems between pattern designers and software developers, allowing include the concept of pattern language that seeks to establish relationships between patterns [7].

**Table 6. Description of Patterns Seffah [7]**

Pattern Identification	
Pattern Name	It describes how the pattern will be Called
Alias:	It describes what the employer receives
Author:	Who designed the pattern?
Category:	Pattern Classification
Keywords:	Allowing be found
Related Patterns:	They can be (Super ordinated, Subordinated Brothers / Neighbors, Competitors)
Context of Use	
User:	Categories of users, people, profiles, etc.
Tasks:	Tasks are structured hierarchically. All sub-tasks must originate from a root.

This table continues on the following page————>



Platform Capacity:	Information should be organized in devices independently.
Problem	
It gives an idea of the problem that the pattern solves. This could be represented as a question	
Forces	
The forces describe aspects of influence of the problem and the solution. This aspect can be represented in a list.	
Solution	
It gives a state of the solution to the problem including the justification of the solution. This should also provide references for further understanding	
Implementation	
Structure	This is a high level of abstraction, by visual modeling notation
Strategy	It includes examples, figures, and sample codes.
Consequences	
Consequences and results of using the pattern. This can be described by a list of metrics, criteria or factors usability.	

On the other hand, Table 7, a table comparing features or elements considered in the definition of the proposed patterns and so far shown reviewed here. It can be seen that all the authors consider the basic aspects that define a pattern such as the problem, the context and the solution. You can identify other issues of great importance such as the name of the pattern, the forces and examples.

**Table 7. Relationship Characteristics Authors**

Characteristics	Authors					
	Tyldwell	Van Welie	Vanderdonckt	Van Duyne	Master Detail	Seffah
Name	X	X		X	X	X
Alias					X	X
Author						X
Classification					X	X
Problem	X		X		X	X
Keywords						X
Solution	X	X	X	X	X	X
Constraints					X	
Forces	X	X	X	X	X	X
Weakness					X	
Justification					X	
Applicability					X	

This table continues on the following page →

Characteristics	Authors					
	Tydwel	Van Welie	Vanderdonckt	Van Duyne	Master Detail	Seffah
Context	X	X	X	X	X	X
Consequences	X	X	X	X	X	X
Structure					X	X
Participants					X	
Contributors						
Implementation					X	X
Example	X				X	
Usability principles	X					
Related Patterns		X		X	X	X

This comparison reveals that the proposals made by Seffah [7] and Master Detail [14] include aspects related to the implementation of standards, coming to consider issues related to structures and implementation strategies. These features are useful when trying to implement the patterns. These two proposals mostly consider the features list above, proving to be the most characteristic feature.

This review identified some important features in describing patterns of interaction. However, in order to corroborate their relevance from the view of other thematic experts in interaction patterns, we designed and implemented a survey form, which is described in greater detail in session 3.

### 3. User experience

Over time different authors have conceptualized the user experience, however, the definition of [16] as “Perceptions and responses or resulting from the use or anticipated use of a product, system or service” is highlighted.

Furthermore, the definition proposed in [17], where user experience is conceptualized as “The feeling, feeling, emotional response, assessment and user satisfaction regarding a product, a result of the phenomenon of interaction with the product and highlights interaction with your provider.”

In [18], as mentioned earlier a review of the main facets that make up the UX, in terms of design and evaluation of products for products with good levels of user experience, provides this proposal is based on the work of different authors (see Figure 1).

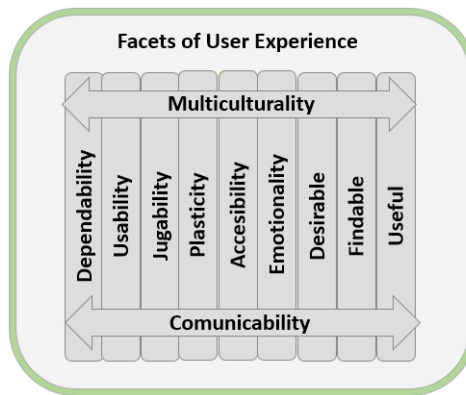


Figure 1. Facets of user experience

The communicability and multiculturalism are transverse to the implementation of other facets displayed vertically aspects. Some of these important aspects are defined to achieve satisfactory user experiences:

### 3.1 Usability

Usability or quality of use, is an anglicized which means ease of use, and whose formal definition refers to the degree of effectiveness, efficiency and satisfaction with which specific users can achieve specific goals in specific contexts of use [19]. In the latest standard [20] defines usability as “the extent to which a product or system can be used by specific users and achieve specific goals with effectiveness, efficiency and satisfaction in a specified context of use.

### 3.2 Emotionality

Emotions are defined as reactions to events related to the needs or goals concerning an individual, including physiological, emotional, behavioral and cognitive components [21, 22]. The perception of emotions in the use of systems is useful for determining user satisfaction and this experience can adjust subsequent developments.

### 3.3 Multiculturalism

In [6] reveal that the software products and general web applications are used by people from cultures for which they are not originally designed, emphasizing the need to base a basis for web designs are multicultural.

### 3.4 Findability

In [5], findability is defined as the measure of the ability of the user to find the information sought within a reasonable time. It is a factor that refers to the possibility of finding or easily retrieve the information needed, a correct result of information architecture, structure, and content description and classification.

### 3.5 Utility

The concept of utility refers to the extent that the website serves the user, as the affective attitude with the website. We must clarify that this factor refers to the subjective or perceived usefulness, not objective or technical [23].

### 3.6 And others

Currently these facets are mostly considered from the evaluation of products, and some of them from the design, resulting in increased efforts of time and resources in product development. In which lies the relevance of this proposal. As we review how to include design aspects of these facets in the pattern description template it is proposed.

Importantly, the revised templates in paragraph III, the only one that includes aspects of UX is that of Jennifer Tidwell, who considers usability in the definition of her proposed standards. Rest not consider these aspects.

## 4. Survey prepared for review of interaction characteristics of patterns

### 4.1 Survey Description

After reviewing the templates, a survey was developed with the objective of identifying the main elements or characteristics used to describe patterns. This survey was applied to thematic experts both in business and academic world. The survey attempts to determine the level of importance of these features, from the views of academics and business with extensive experience in these area professionals, considering their different views and opinions.

The hypothesis on the development of this survey is aimed at verifying what features to consider in describing patterns of interaction. The questions are open and closed type,



and defined variables are: time experience in the field and the level of importance of each of the features described in Table 1.

The distribution of the survey was carried out using the free Google tool to create forms Google Form. A sample of 35 answered surveys was obtained with a sensitivity of 5%. Although the contact through the Internet was fast, wait times in the responses have been slow. However, the implementation of the survey was achieved successfully.

## 4.2 Analysis of Results

The findings after analyzing the results of the application of the survey are described below:

- Respondents are academics and the business sector with experience in interaction patterns mostly between 5 and 10 years. For some cases, it was possible to survey people over 10 years of experience
- Aspects such as classification, problem, solution and examples have been chosen as very important by 100% of respondents.
- In the case of other aspects, such as the pattern name, restrictions, applicability and consequences they were selected high importance by 80 % of respondents.
- Other aspects such as strength, context, structure, implementation and related patterns have been chosen as very important by 60% of respondents.
- The level of importance for the following characteristics was half weakness, participants and collaborators.
- Alias was a minor feature.
- In relation to include other features in describing patterns only 15% of respondents said yes, making suggestions such as include aspects of design decisions and the context should be broader, for example device or platform, environment work (organizational unit).

## 4.3 Choice of the template

Whereas the elements identified in the conceptual review and the results of the survey, the template was defined for describing patterns of interaction to be proposed in this research.

Aspects such as the problem and the solution are considered key elements in both languages checked patterns, as in the answers of respondents. Another important element is the examples to verify the success of the patterns. In the case of context and despite not being considered of high importance for all respondents, is a fundamental characteristic among these authors, because a mostly agree to describe the categories, users and challenges of their patterns.

Other aspects are also featured: the forces, pattern name, restrictions, consequences and related patterns. Not considered in its entirety in the proposed interaction patterns, or inquiry with experts, but representing high importance when defining patterns.

Among the most representative examples are those of Seffah [7], Master Detail [15] and Tydwell [12], in relation to the number and significance of the criteria considered, however after evaluating these criteria to experts, the proposal that comes closest is the Sefah [7], considering that includes aspects of implementation. Although Master Detail [15] turns out to be a complete, to inquire among experts as the structure, justification, the applicability proposal, Tidwell's [12] proposal does not include aspects of implementation.

According to these findings, it is considered using the template describing patterns Seffah proposal [7] with some variations. This template comes together most aspects identified as relevant here both by the authors of the languages of interaction as respondents, especially as a proposal that includes implementation issues and relationships between patterns, the latter of great importance in creating a language of interaction patterns.

Regarding suggestions broader context, including aspects related to the types of devices and platforms, and the workplace is identified that these aspects have already been considered in the proposal Seffah [7] from defining the context.

Another aspect identified as of great importance, not included in the proposed Seffah [7] corresponds to the restrictions, which provide much information as to the limitations of the pattern, so it was decided to include them.

The review of facets of the UX has led to consider these aspects in the template description of the patterns of interaction. To this end, they have been reviewed and evaluated some guidelines or design of these facets (see Figure 2) and has made inclusion in the template.



Figure 2. Example Implementation Strategy

So far, they have reviewed aspects of multiculturalism, accessibility and usability, but it is necessary to continue reviewing design guidelines to consider other facets included in the definition of the patterns. In the case of multiculturalism, we have reviewed the proposed work on [24], for accessibility the W3C guidelines for accessibility and in the case of usability, the Guidelines Nielsen.

Considering the above, in Table 8 shows an example, the template description of patterns that were used in this research. The example is a proposal of a pattern of interaction for mobile devices that allows you to search for information and get a result set.

Table 8. Adaptation Template description Seffah [7]

Pattern identification	
Pattern name:	App Search Results
Alias:	Search Results
Author:	Yuliana P, César C, Josefina G,
Category:	Interaction Patterns
Keywords:	Content search
Related Patterns:	Related
	Competitors
	Superordinate
	Subordinate
	Neighbors      Search_Area, Search_Box Visualizer

This table continues on the following page →

Context of Use	
User:	Smartphone user technologies.
Tasks:	It is necessary to conduct a search for information, review the results of the search, omit or select one in particular.
Platform Capacity:	Smartphone operating system IOS, Android.
Problem	
Users need to ask a question and get a result set.	
Forces	
The user need a brief description of the results with highlighted keywords search. The user must know first closest to the search results, and thus in order of importance.	
User Experience Aspects	
Usability Aspects	They are implemented when its hypertext structure exceeds 150. They should introduce a standard form. The size of the box should be wide enough to allow the user to enter multiple keywords.
Accessibility Aspects	The pages should have a title that describes its subject or purpose, indicating the language, there should be links with the same description and different destinations, links to related pages, a table of contents or site map.
Multiculturalism Aspects	The buttons and links should clearly indicate what action will be taken.
Solution:	
A control that allows you to display the results of the search, categorize search results, results show special is used.	
Constraints:	
It must be validated connection to the database. It should validate data integrity.	
Implementation:	
Structure:	Receives a corresponding string with the search, the class is watching the event and generates search returning a list of objects corresponding to the display pattern.
Strategy	In this example a search for an application on the Appstore is made and generates a list of answers. Which are deployed or displayed on the screen, see Figure 3.
Consequences	
Search processes are optimized. The results are organized and categorized, greater ease in selecting results	

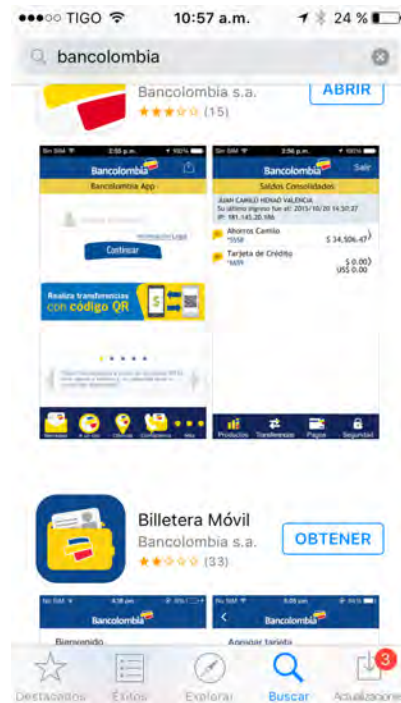


Figure 3. Example Implementation Strategy

## 5. Conclusions and Future Work

Although there is no consensus among the authors of patterns and pattern languages interaction about which elements to consider, and that each author chooses freely about these features, it is worth noting that each of them to a greater or lesser importance represents an element of judgment to define or characterize the pattern.

When it comes to implementing interaction patterns it is important to consider implementing elements from the definition thereof. On the other hand, when it comes to creating interaction pattern languages it is important to consider their interrelationships as a feature in the definition or description of the patterns.

After defining the template for describing patterns of interaction, this proposal gives way to design interaction patterns proposed for the language patterns of interaction focused on user experience, the main objective of this research, then automate the process interfaces generation from these patterns ensuring the inclusion of user experience from design.

## 6. Acknowledgment

In the development of this work we appreciate specialists in HCI, academics and professionals who assisted in filling out the survey.

## 7. References

- [1] Van Welie, M., & Troetterbeg, H. (2000). Interaction Patterns in user Interfaces. PLoP 2000.
- [2] [Bayle, E., Bellamy, R., & Casady, G. (1998). Putting It All Together: SIGCHI, 17-24.
- [3] Caceres, J. (10 de Mayo de 2008). Patrones de diseño: ejemplo de aplicación en los Generative Learning Object Design patterns: example of application in the Generative Learning Object. Retrieved 10 de Noviembre de 2015 from Revista de Educación a Distancia: <http://www.um.es/ead/red/M10/caceres.pdf>.
- [4] Arhippainen, L., & Tahti, M. (2003). Nosolousabilidad. Retrieved octubre de 2015 from Nosolousabilidad: [www.nosolousabilidad.com/articulos/experienciadeusuario.htm](http://www.nosolousabilidad.com/articulos/experienciadeusuario.htm)
- [5] Morville, P. (2006). Information Architecture for the World Wide Web: Designing Large-Scale Web Sites (Vol. 3). O'Reilly Media.
- [6] Masip, L., Gil, R., Granollers, T., & Collazos, C. (2009). Multiculturalidad e internacionalización en interfaces Web Revista Avances en Sistemas e Informática. Revista Avances en Sistemas e informática, 6 (1657-7663), 191-196.
- [7] Seffah, A. (2015). Patterns of HCI Design and HCI Design of Patterns. Human-Computer Interaction Series, Bridging HCI Design and Model-Driven Software Engineering, A. Seffah, DOI 10.1007/978-3-319-15687-3\_2, © Springer International Publishing Switzerland 15 ISBN 978-3-319-15686-6.
- [8] Muñoz, J., & Rodríguez, G. (2008). Patrones de Interacción: Una Solución para el Diseño de la Retroalimentación Visual de Sistemas Interactivos. Instituto Nacional de Astrofísica Óptica y Electrónica (INAOE), Departamento de ciencias computacionales. Puebla: Instituto Nacional de Astrofísica Óptica y Electrónica (INAOE).
- [9] Buschmann, Frank; Henney, Kevin; Schmid, Douglas. Pattern-Oriented Software Architecture, Volume 4: A Pattern Language for Distributed Computing, Wiley, 2007.
- [10] Alexander, C. (1979). A Pattern Language. Center for Environmental Structure, Berkley California.
- [11] Gamma, E., & Helm, R. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Addison Wesley Professional.
- [12] Tidwell, J. (2011). Designing Interfaces (Vol. 2). O'Reilly Media.
- [13] Van Duyne, Douglas, Landay, James and Hong Jason. Design of Sites: Pattern Language for Web, 2002. Pearson Education.



- [14] Thanh-Diane Nguyen, Jean Vanderdonckt, Ahmed Seffah, Generative Patterns for Cross-Platform User Interfaces Engineering: The Case of the Master Detail Pattern. 2015 Louvain School of Management.
- [15] van Welie, M., van der Veer, G., & Eliens, A. (2000). Patterns as Tools for User Interface Design. Amstendarnd.
- [16] Bevan, N. (2008a). UX, Usability and ISO Standards. London: W3 9RG.
- [17] Bevan, N. (June de 2008b). Classifying and selecting UX and Usability measures. COST294-MAUSE Workshop: Meaningful Measures: Valid Useful User Experience Measurement.
- [18] Masip, L. A. (2013). User experience methodology for the design and evaluation of interactive systems. Lleida: Universidad de Lleida.
- [19] ISO. (2009). ISO FDIS 9241-210.: ISO.
- [20] ISO/IEC 25010. Systems and software Engineering-Systems and software Quality Requirements and Evaluation (SQuaRE) System and software quality models.ISO
- [21] Brave, S., & Nass, C. (2003). Emotion in Human-Computer Interaction. In The human-computer interaction handbook (pp. 81-96). NJ, Hillsdale, USA: L. Erlbaum Associates Inc.
- [22] Méndez, Y., Collazos, C., & Granollers, T. (2014). Evaluating Interactive Systems from an Emotional Perspective. Revista Científica Guillermo de Ockam, 43-49.
- [23] Montero, H. (2006). Factores de Diseño Web Orientados a la Satisfacción y no frustración de Uso. Revista Española de Documentación Científica, 239-257.
- [24] Zapata V, Palacios A., Colazos C., Muñoz J., Alvarez F., Silva A. MULTICULTURALISM PATTERNS FOR WEB APPLICATION DESIGN Lámpsakos | No.11 | pp. 19-28 | enero-junio | 2014 | ISSN: 2145-4086 | Medellín – Colombia.

# Chapter # 2

## Using ISO/IEC 29110 Deployment Package to construct educational video games in software engineering

Eréndira M. Jiménez-  
Hernández, Hanna  
Oktaba, Frida Díaz-  
Barriga Arceo  
National Autonomous  
University of Mexico  
Ciudad de México, México  
{erendira.jimenez, hanna.  
oktaba} @ciencias.unam.  
mx, fdba@unam.mx

Mario Piattini  
University of Castilla-La  
Mancha  
Ciudad Real, España  
mario.piattini@uclm.es

Alan M.  
Revillagigedo-Tulais,  
Daniel Barcenás-Acosta  
Arturo López-Guzmán  
Sergio V. Flores-Zarco  
Morelia Institute of  
Technology  
Morelia, México  
{alanmarth, barcenan.dan,  
arturolg01}@gmail.com,  
vladi\_flores@hotmail.com

### 1. Introduction

Educational video games are an interactive, attractive and entertaining technology built around identities that work with good learning principles [1]. Some of their learning principles are: active learning [2], semiotic domain [3] and metacognition [4].

It exists some proposals of educational video games in Software Engineering such as: [5] designed for early programming education, [6] constructed to encourage collaborative behavior in teams, [7] developed to teach concepts of software engineering, [8] constructed to facilitate the requirements elicitation, [9] designed to learn key concepts of object-oriented design patterns and [10] to teach the software engineering process.

Developing educational video games is a complicated task that involves the expertise of professionals from various disciplines including Computer science, graphic design and Pedagogy [11].

This chapter presents a methodology created from the ISO/IEC 29110 [12] standard for Deployment Package, to construct educational video games in software engineering. Our methodology helped us to construct one video game named Alphaspot [13], which was

designed to facilitate the learning of the Essence kernel [14] to 12 practitioners that work in a software enterprise.

The chapter is organized as follows: Section II explains the pedagogical content of the developed educational video game: the Essence kernel.

Section III presents our proposal of methodology to construct educational video games in software engineering.

The constructed educational video game and its experience evaluation are presented in Section IV as results of the use of the methodology.

Finally, our conclusions and future work are detailed in Section V.

## 2. Essence

Essence is a standard approved in 2014 by the Object Management Group (OMG). It is constituted by methods, practices, one kernel and one language.

The Essence kernel includes the empirical knowledge of software engineering. It is a framework of thought that permits the reasoning about the progress and health of a software project. It comprises three areas of concern: costumer, solution and endeavor. Each area of concern has a set of Alphas (essential things to work with), a set of Activity Spaces (essential things to do) and Competencies (essential capabilities required) in a software project.

As shows Figure 1 the Essence kernel has seven Alphas: opportunity, stakeholders, requirements, software system, team, work and way of working. Each Alpha has a set of states sequentially interrelated, and each state has a checklist.

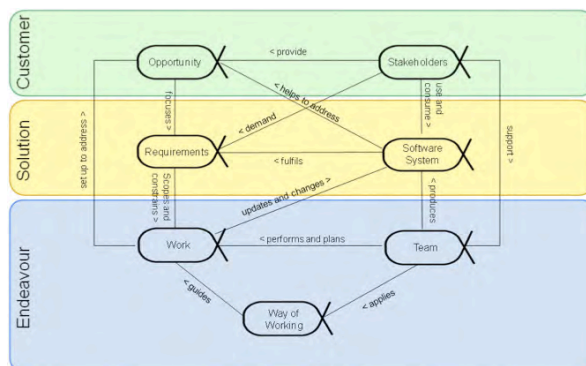


Figure 1. The Alphas of Essence [14]

### 3. Methodology to construct educational video games

The methodology to construct educational videogames is composed by three stages and three processes. The stages are: pre-production, production and post-production, as specifies in [12, 15]. The processes are: project management, software implementation and pedagogical implementation (see Figure 2).

In order to create the activities for each process, it is necessary to identify three main actors: for the “project management” process it is desirable a “Project Manager/leader” (PRM), for the “software implementation” process it is necessary a “video game development team” (VGDT), and for the “pedagogical implementation” process it must to exist a “Pedagogic Manager”(PEM).

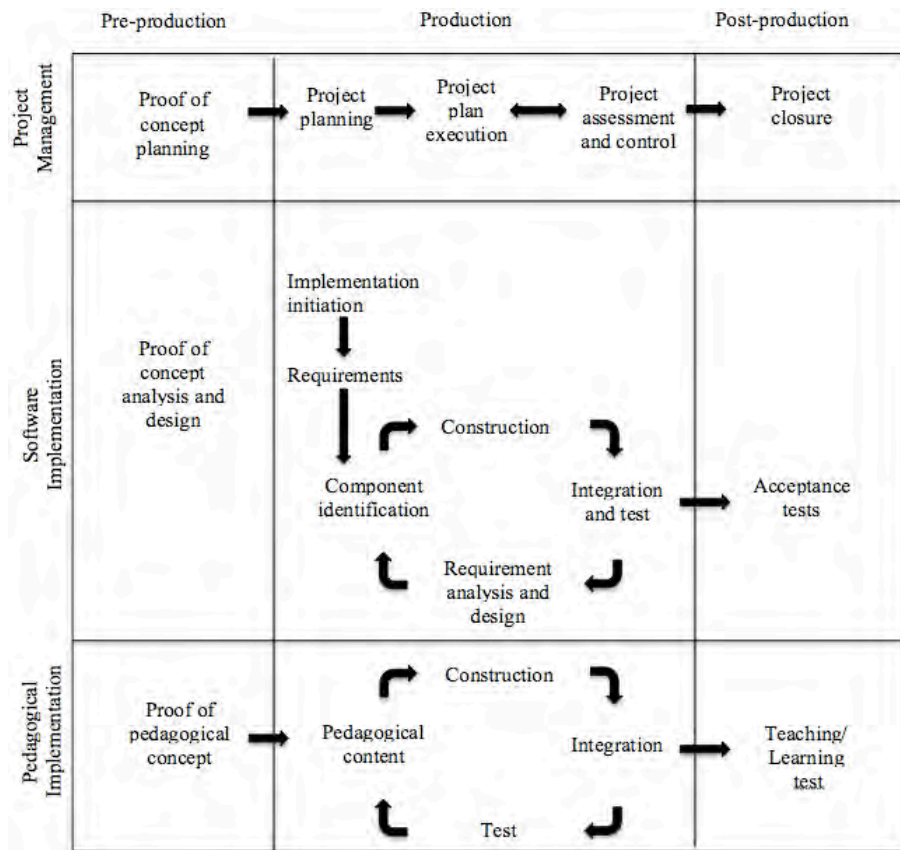


Figure 2. Methodology to construct educational video games

In the pre-production, the activities included are:

- a. Define the objective of learning (by PEM).
- b. Search information related to the aim of learning (by PEM).
- c. Select the educational content (by PEM).
- d. Share/teach the educational content [16] with the members of the video games development team (by PEM).
- e. Define the context/scenario of the video game according with the educational content (by VGDT).
- f. Propose the storyboard of the educational video game (by VGDT).
- g. Design the characters of the educational video game (by VGDT).
- h. Verify the design of the educational content (by PEM).
- i. Obtain/specify the educational video game requirements (by PRM).
- j. Make the documentation of the educational video game designs (by PRM).
- k. Estimate costs (by PRM).
- a. Define the project plan (by PRM).

In the production, the activities included are:

- a. Codify the video game (by VGDT).
- b. Verify the implementation of the educational content (by PEM).
- c. Ensure compliance with the project plan (It can make use of the kernel Alphas of Essence) (by PRM).
- d. Ensure the quality of the construction processes of the video game (by PRM).

In the post-production, the activities included are:

- a. Confirm that the video game fulfills the learning objective (by PEM).
- b. Test the video game (by VGDT).
- c. Assess the video game quality (by PRM).
- d. Make the final documentation of the educational video game (by VGDT and PRM).

As part of the foundation of the information, it is necessary to create 4 documents:

- a. Pedagogical document. It describes: (1) teaching objective, (2) learning level, (3) interaction scheme, (4) synthesis of information through glossaries, mind maps, synoptic maps and/or conceptual paintings and (5) test with questions and answers to evaluate the learning. This document is the result to perform the activities a) to d).

- b. Document of definition of the project. It describes: (1) name of the educational video game, (2) aim of the educational video game, (3) features of the educational video game, e.g. Target audience, languages, operating systems, etc., (4) technologies, e.g. Programming languages, game engine, modeling software, etc., (5) video game history, (6) storyboard, (7) cards of the characters with name, features, actions, etc., (8) list of requirements, (9) UML diagrams [17] and (10) Play-flow diagrams [18]. This document is the result to perform the activities e) to j).
- c. Project schedule document. It describes: (1) tasks list, (2) assignment of responsibilities and (3) delivery dates. This document is the result to perform the activities k) and l).
- d. Final document of the project. It describes: (1) general project information, e.g. Project's name, date, authors, path/address of the code, executable project and documents A-C, etc., (2) configuration management, (3) game overview, e.g. name, objective, story and look and feel of the educational game video, (4) game play, e.g. play flow, characters, assets, etc., (5) artificial intelligence, e.g. collision detection, (6) technical information, (7) user manual, (8) Appendices, e.g. script of voices, message list, commented code, etc. This document is the result to perform the activities m) to t).

The processes of Project Management and Software Implementation could result familiar in software engineering, but the Pedagogical Implementation process not that much. It is the reason why we propose conducting a techno-pedagogical design based on constructivist learning environments, such as [19]. The aim of the Jonassen's Model is to promote problem solving and conceptual development [20]. It allows designing environments that involve the members of the construction team in the development of knowledge [21].

Jonassen explains that the essential components (see Figure.3) in the constructivist learning environments include: (1) question or project as the focus of the environment, it implies the context, the simulation and the manipulation space problems; (2) related cases which provide different perspectives; (3) information resources, it implies to provide just-in-time information to help learners comprehend and solve the problem; (4) cognitive tools such as computer tools that help to visualize, organize, automate, or supplant thinking skills; (5) conversation/collaboration tools such as e-mails, Skype® and (6) social/contextual tools such as reflexive diary.

We compare our methodology with [22]. It has 7 stages. Experts in the field of education carry out the first and the second stages. The artistic director and scenarist conduct the next three stages. The sixth stage is the construction of the game, and the last one consists in the documentation of the game.

In our methodology, we have 3 stages, and each one of them has 3 connected processes. However, even though the methodology proposed has fewer stages; we accomplish the same objective.

Also, we compare our proposal with MISA [23], which has five stages: analysis of the preliminary design, elaboration of architecture, the design of educational materials, realization of the pedagogic material and validation of pedagogic material. In our proposal, we grouped the three first stages of MISA in the “pre-production” stage. The fourth stage of MISA corresponds to “production” stage. And the last stage is the “post-production” in our methodology.

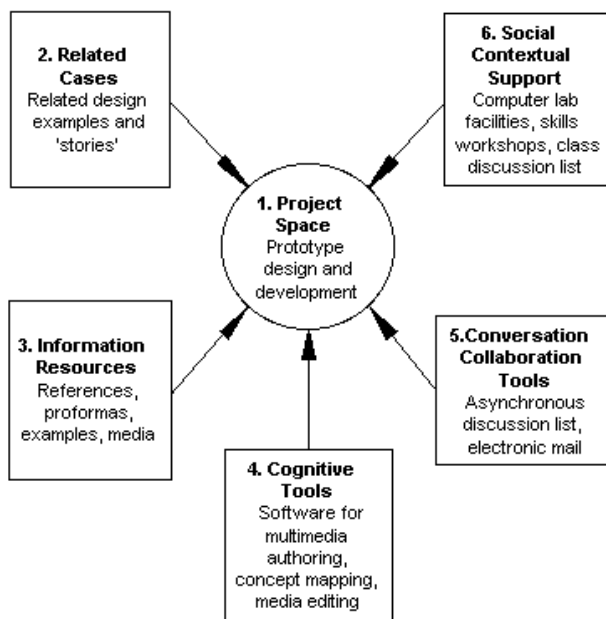


Figure 3. Jonassen's Model

## 4. Results

One educational video game named “Alphaspot” was constructed using the methodology proposed in this chapter. To program Alphaspot, we employed the game engine “Unity 5” [27] and the program languages C# and JavaScript.

It can be played for a single player [24]. With this video game it is possible to obtain the learning levels of remembering and understand [25] through the four sensorial styles: visual, auditory, kinesthetic and reading/writing [26].

The Alphaspot's objective is to facilitate the learning of the Alphas of Essence and its states. 41 levels constitute this educational video game (one for each state of each Alpha).

The name of the main character of this educational video game is Alphaspot (see Figure 4).

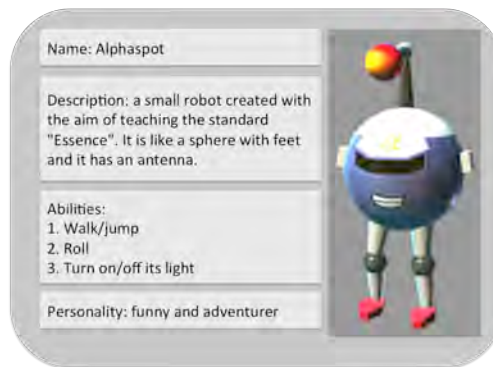


Figure 4. Card of the main character

Alphaspot is a sphere that can walk, jump, roll, turn on/off its antenna and change the color of its antenna. Alphaspot and the other characters (the enemies, alphaspirts and the final boss) were modeled using 3Dmax Studio®, Adobe Photoshop® and Adobe Flash® (see Figure 5).



Figure 5. Alphaspot's model



Alphaspot starts with an initial menu (see Figure 6), it appears an initial menu, which has four options: new game, load game (for games previously played), credits and exit.

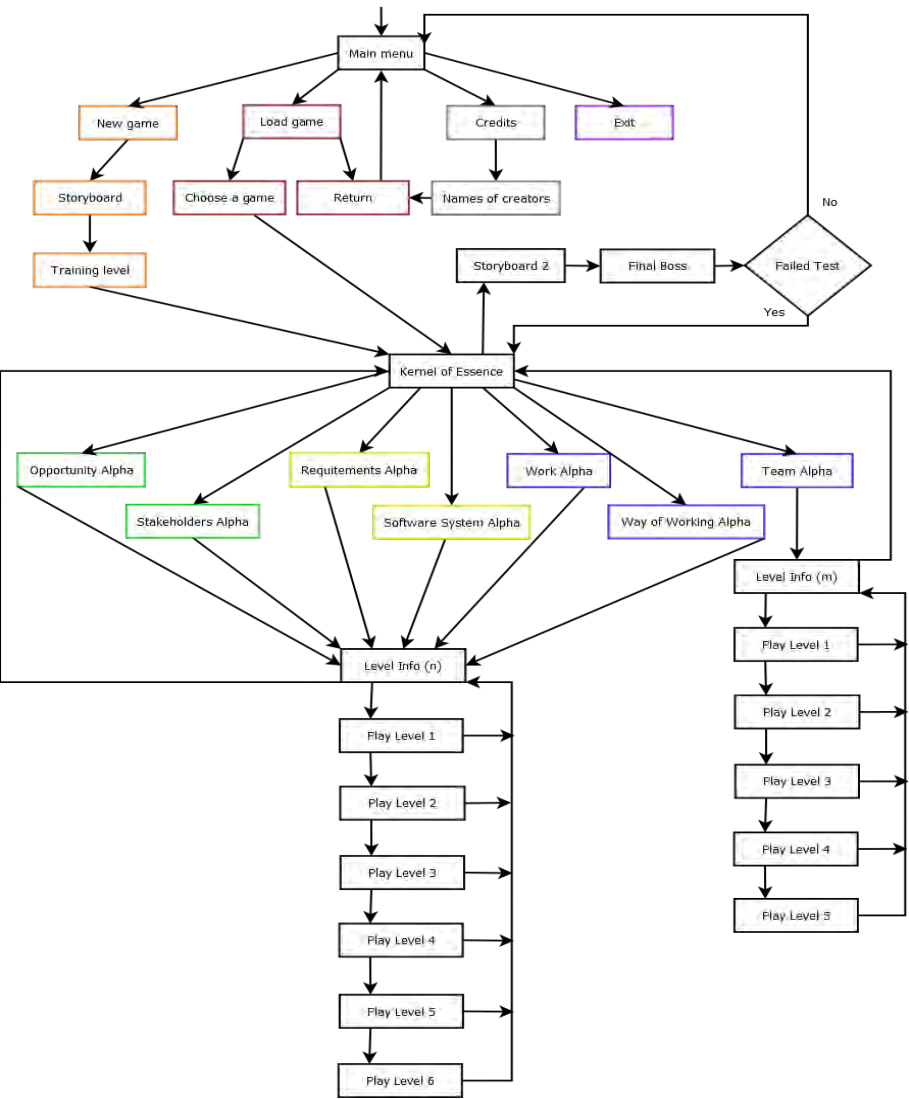


Figure 6. Alphaspot's play flow

If the player chooses “new game”, it appears a video that explains the context and the history of the video game: in a meeting of one software development team, there is a

discussion, in which all the members are blaming ones each other for the problems of their software project. Suddenly, the main character of the video game (Alphaspot) gets inside the room and starts to explain them what is Essence and how can they use it to solve the problems. The team decides to use Essence, but Alphaspot explains that its enemies have robbed the state cards. So, the player needs to find the 41 cards in the video game to comply the mission and to help to the software development team (see Figure 7a).

After the introductory video, the player is introduced in a training level (see Figure 7b) where s/he can learn to play. When the player passed this level, s/he can see the main menu (see Figure 7c) that represents the seven Alphas with portals. Each portal has a color (green, yellow or blue) that permits to identify its area of concern (costumer, solution and endeavor). The player can choose whatever Alpha because there is not a specific order. When the player moves to Alphaspot next to one portal, there is a character named “Alphaspirit” that brings all the information about the corresponding Alpha (see Figure 7d).

When the player gets inside to one Alpha, s/he will see the states of the Alpha representing by levels. The levels of one Alpha are sequential (see Figure 7e).

When the player gets inside to one level, s/he needs to find all the checkpoints of the state to obtain its card. E.g. Figure 7f shows the screen of the first level called “Principles established”, that belongs to Alpha “Way of Working”, here the player needs to find the eight checkpoints to get the card. Each level has a different appearance, sound and enemies (see Figure 7g).

Once the 41 cards are collected, the player will see a video that will show the characters in the initial meeting but now using the Alphas cards to solve their problems (see Figure 7h).

Finally, the video game has an evaluation level, in which a “final boss” makes some questions to Alphaspot, and he has three possible options for choose one, as shown in Figure 7i.

Alphaspot was developed in seven weeks. It is available in two languages: English and Spanish. It can be executing on computers with Mac OS® and Windows®. It can be downloaded from [www.alphaspot.com.mx](http://www.alphaspot.com.mx) and from Google Play. Figure 8 shows the adaptation of the video game to one mobile with Android.



(a)



(b)



(c)



(d)



(e)



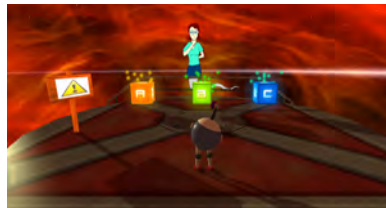
(f)



(g)



(h)



(i)

Figure 7. Alphaspot's screens: (a) Initial meeting, (b) Training level, (c) Main menu, (d) Alphaspot next to one portal, (e) Levels of the Alpha "Requirements", (f) Level "Principles Established" of Alpha "Way of working", (g) Level "Seeded" of Alpha "Team", (h) Final meeting, (i) Evaluation level.



Figure 8. Alphaspot in Android

We evaluated Alphaspot with the IMI (Intrinsic Motivation Inventory) as did in [28]. IMI is a method to evaluate the perceived experience when certain persons are performing some activity. IMI has 36 questions that evaluate 7 factors: (1) interest/enjoyment, (2) perceived competence, (3) effort, (4) pressure/stress, (5) perceived choice, (6) value/usefulness and (7) connection. It has an answer scale from 1 to 7, where 1 represents “not completely true” and 7 means “very true”.

We randomly selected 12 practitioners in a Mexican software entity to play Alphaspot per 2 hours for 4 days. The Spanish version of Alphaspot for computers was used in the experiment.

After the 8 hours played, all the practitioners obtained the 41 cards, and eleven of the twelve practitioners approved the final test (the evaluation knowledge level).

Then, the practitioners answered the 36 questions of the IMI. Table 1 shows the average of each practitioner by each factor.

The results of the questionnaire shows that: For the IMI’s factor 1 the average was 6.5, it means that the practitioners enjoyed to play with Alphaspot; for the IMI’s factor 2 the average was 5.9, it means that the practitioners felt that they had the competencies needed to play Alphaspot; for the IMI’s factor 3 the average was 3.7, it could be interpreting as the practitioners did not feel much effort playing with Alphaspot; for the IMI’s factor 4 the average was 1.7, it implies that they did not feel much pressure or stress playing Alphaspot; for the IMI’s factor 5 the average was 5.9, which means that the practitioners felt that they could choose what they wanted to learn and keep their own rhythm; for the IMI’s factor 6 the average was 6.4, it means that the practitioners thought that Alphaspot was usefulness; and finally, for the IMI’s factor 7 the average was 6.3, so, the practitioners felt connection with the educational video game.

TABLE I. Results of IMI

Pract.	FACTOR						
	1	2	3	4	5	6	7
1	7	7	4.5	2	7	7	7
2	6	5	2.3	1	5	6	6
3	7	7	6.8	1	7	7	7
4	7	7	2.5	1	7	7	7
5	5	6	7	2.5	4	5.8	5
6	6.5	4	3	1.7	6	6.2	6
7	7	7	2.5	1	7	7	7
8	6.2	3	4.5	2.8	5	5.6	6
9	5.8	5	2.8	3	3	5.2	5
10	7	7	2.5	1	7	7	7
11	6	6	4.7	2	6	5.8	6
12	7	7	1.5	1	7	7	7
Average	6.5	5.9	3.7	1.7	5.9	6.4	6.3

## 5. Conclusion and future work

This chapter presented an adaptation of the ISO/IEC 29110 standard for “Deployment Package”, to develop educational video games in the area of software engineering.

The proposed methodology has three stages: pre-production, production and post-production. In addition, it has three processes: project management, software implementation and pedagogical implementation.

To use the methodology, it is necessary to have three types of roles: project manager/leader, pedagogic manager and programmers.

As a result of the usage of the methodology, one educational video game named “Alphaspot” was created to facilitate the learning of Alphas of the Essence kernel. The Alphas allow assessing the health and progress of a software effort.

Twelve practitioners working in a software entity played Alphaspot. They answered 36 questions to evaluate the perceived experience after play the educational video game.

With the obtained results, we will improve some ludic aspects of Alphaspot and we will test the game again with more people.

We are planning to construct more educational video games in order to keep testing our methodology.

## 6. Acknowledgment

This work has been funded by the Graduate Science and Engineering Computing (UNAM) and the CONACyT grant scholarship program.

## 7. References

- [1] UNESCO, "El niño y el juego. Planteamientos teóricos y aplicaciones pedagógicas", Estudios y documentos de educación vol. 34. París, France, pp. 1-20, 1980.
- [2] Meyers, C. and Jones, T., "Promoting Active Learning: Strategies for the College Classroom". San Francisco, USA: Jossey-Bass, pp.1-24, 1993.
- [3] Gee, J. P., "What video games have to teach us about learning and literacy, revised and updated". Basingstoke: Palgrave Macmillan, pp. 5-16, 2008.
- [4] Flavell, J. H., "El desarrollo cognitivo". Madrid, España: Visor, pp. 24-239, 1993.
- [5] Paliokas, I., Arapidis, C. and Mpimpitsos, M., "PlayLOGO 3D: A 3D interactive video game for early programming education". Proc. Third International Conference on Games and Virtual Worlds for Serious Applications, pp. 24-31, 2011.
- [6] Wendel, V., Gutjahr, M., Göbel, S. and Steinmetz, R., "Designing collaborative multiplayer serious games. Escape from Wilson Island - A multiplayer 3D serious game for collaborative learning in teams". Education and Information Technology, vol. 18, pp. 287-308, 2013.
- [7] Cooper, K. and Longstreet, C., "Towards model-driven game engineering for serious educational games: Tailored use cases for game requirements". Proc. of the 17th International Conference on Computer Games, (CGames '12), pp. 208-212, 2012.
- [8] Shabalina, O., Sadovnikova, N. and Kravets, A., "Methodology of Teaching Software Engineering: Game-based Learning Cycle". Proc. of the Third Eastern European Regional Conference on the Engineering of Computer Based Systems, pp.113-119, 2013.
- [9] Rusu, A. Russell, R. Cocco, R. and DiNicolantonio, S., "Introducing Object Oriented Design Patterns through a Puzzle-Based Serious Computer Game". Proc. of the 41st ASEE/IEEE Frontiers in Education Conference, (FIE '11), pp. 1-6, 2011.
- [10] Zhu, Q., Wang, T. and Tan, S., "Adapting Game Technology to Support Software Engineering Process Teaching: From SimSE to MO-SEProcess". Proc. of the Third International Conference on Natural Computation, (ICNC '07), pp. 777-780, 2007.
- [11] Tavinor, G., "The art of video games". Wiley-Blackwell, pp. 20-34, 2009.
- [12] ISO/IEC 29110, "Deployment Package. Part 1- Technical Description". International Organization for Standardization (ISO), pp. 1-48, 2015.

- [13] Jiménez-Hernández, E., Oktaba, H., Revillagigedo-Tulais, A., Flores-Zarco, V., Barceñas-Acosta, D. and Guzmán-López, A., "Alphaspot". Retrieved from: [www.alphaspot.com.mx](http://www.alphaspot.com.mx), 2015.
- [14] OMG, "Kernel and Language for Software Engineering Methods (Essence)". USA:OMG, pp. 15-62. Retrieved from: <http://www.omg.org/spec/Essence/1.0/>, 2014.
- [15] Jimenez-Hernández, E., Oktaba, H., Piattini, M., Díaz-Barriga, F., Revillagigedo-Tulais, A. and Flores-Zarco, V., "Methodology to construct educational video games in software engineering". Proc. of the 4th International Conference in Software Engineering Research and Innovation (CONISOFT '16), pp. 110-114. doi: 10.1109/CONISOFT.2016.25, 2016.
- [16] Jonassen, D. y Rorher-Murphy, L., "Activity Theory as a framework for designing constructivist learning environments". Educational Technology: Research and Development, vol. 46 no.1, 1999.
- [17] Booch, G., Rumbaugh, J. and Jacobson, I., "The Unified Modeling Language User Guide". Addison Wesley Longman, 1998.
- [18] Arrabales-Moreno, R., "Desarrollo de la lógica de un videojuego". Universidad Carlos III Madrid, pp. 35-36, 2012.
- [19] Jonassen, D., "El diseño de entornos constructivistas de aprendizaje". Aula XXI Santillana, pp. 93-224, 2000.
- [20] Jonassen, D., "Evaluating Constructivist Learning". Educational Technology, vol. 31, no.9, (ERIC Document Reproduction Service No. EJ433315), pp. 28-33, 1991.
- [21] Jonassen, D. & Rorher-Murphy, L., "Activity Theory as a framework for designing constructivist learning environments". Educational Technology Research and Development, vol. 46, no.1, doi: 10.1007/BF02299477, pp. 61-79, 1999.
- [22] Yusoff, A., Crowder, R., Gilbert, L. and Wills, G., "A Conceptual Framework for Serious Games". Proc. of the Ninth IEEE International Conference on Advanced Learning Technologies, pp. 21-23, 2009.
- [23] Paquette G., Crevier F., Aubin C., "Méthode d'ingénierie d'un système d'apprentissage (MISA)". Revue Informations In Cognito, vol. 8, pp. 37-52, 1997.
- [24] Wenger, E., "Cultivating Communities of Practice (Hardcover)", USA: Harvard Business Press, pp. 1-45, 2002.
- [25] Anderson, L. & Krathwohl, D., "A Taxonomy for learning, teaching and assessing: a revision of Bloom's Taxonomy of educational objectives". New York, USA: Addison Wesley Longman, pp. 1-82, 2001.
- [26] Hawk, F. and Shah, A., "Using Learning Style Instruments to Enhance Student Learning". Decision Sciences Journal of Innovative Education vol. 5, pp. 1- 19, 2007.
- [27] Unity, "Create and connect with Unity 5". Retrieved from: <https://unity3d.com/es>, 2015
- [28] Yamabe, T. and Nakajima, T., "Playful training with augmented reality games: case studies towards reality-oriented system design". Multimedia Tools and Applications, vol. 62, pp. 259-286, 2013.

# Chapter # 3

## Associating quality measures to the alpha states of the SEMAT kernel

**Carlos Mario Zapata Jaramillo**  
Departamento de Ciencias de la  
Computación y de la Decisión  
Universidad Nacional de Colombia  
Sede Medellín  
Medellín, Colombia  
cmzapata@unal.edu.co

**Yury Montoya Pérez**  
Estudiante de Maestría en Ingeniería de  
Sistemas  
Universidad Nacional de Colombia  
Sede Medellín  
Medellín, Colombia  
ymontoyap@unal.edu.co

### 1. Introduction

The ISO/IEC 25000 (also called SQuaRE) standard provides comprehensive coverage to software quality [2]. Such a standard was developed in order to cover two main processes: “software quality requirements specification and systems and software quality evaluation” based on a systems and software quality measurement process [2]. ISO/IEC 25000 standard comprises five main divisions: 1) quality management division; 2) quality model division; 3) quality measurements division; 4) quality requirements division; and 5) quality evaluation division. ISO/IEC 25000 standard replaces the ISO/IEC 9126 [1] and the ISO/IEC 14598 standards [1]. Thus, the ISO/IEC 9126 standard is the foundation for the development of the SQuaRE series [4].

Nowadays, the lack of a theoretical basis and the prevalence of fads and fashions are some of the challenging issues of software engineering [4]. SEMAT (Software Engineering Method and Theory) has been proposed for addressing such issues [5]. “SEMAT is an initiative developed for re-founding the software engineering by defining theoretical basis, best practices, and a set of widely-agreed elements” [6, 7]. Alphas are some of the elements of the SEMAT kernel and they can be used for assessing the health and progress of a software engineering endeavor [5]. Seven alphas are included in the SEMAT kernel. Alphas are representations of the essential things to work with [7].



According to the ISO/IEC 25000 standard, a measure is defined as “a variable to which a value is assigned as the result of measurement.” [2] In previous work [18], we use the ISO/IEC 9126 metrics for measuring the states of some SEMAT kernel alphas. In this Chapter, we propose the selection of the appropriate measures of the ISO/IEC 25023 standard for evaluating and validating some alpha states of two SEMAT kernel alphas—requirements and software system—in order to obtain a high-quality software product.

This Chapter is organized as follows: in Section 2 we review the theoretical framework, which includes an introduction to the ISO/IEC 25000 and ISO/IEC 25023 standards for systems and software quality evaluation, an introduction to the ISO/IEC 9126 standard for software product quality, and an overview of the Software Engineering Method and Theory. The state of the art about ISO/IEC 25023 measures related to the alphas software system and requirements is presented in Section 3. In Section 4, the relationship between ISO/IEC 25023 and the alpha states of the SEMAT kernel is carried out. Finally, in Section 5 conclusions and future work are discussed.

## 2. Theoretical framework

### 2.1 ISO/IEC 25000—SQuaRE

ISO/IEC 25000 was developed in order to address “software quality requirements specification and systems and software quality evaluation” [2] and comprises five divisions:

- » ISO/IEC 2500n—Quality Management Division
- » ISO/IEC 2501n—Quality Model Division
- » ISO/IEC 2502n—Quality Measurement Division
- » ISO/IEC 2503n—Quality Requirements Division
- » ISO/IEC 2504n—Quality Evaluation Division

SQuaRE is aimed to support the development of software products with “the specification and evaluation of quality requirements” [2]. First division includes terms and definitions, and the models of the SQuaRE series. Also, this division can be used to provide guidance for managing product requirements specification and evaluation [2]. Second division includes “detailed quality models for systems and software product, quality in use, and data” [2]. Third division includes internal, external and quality-in-use measures. Also, this division includes a mathematical definition of the quality measures and guidance in order to use them [2] by referencing the ISO/IEC 9126-2, -3, and -4 [8]. Fourth division is intended to help to specify quality requirements. Fifth division includes “requirements, recommendations and guidelines for product evaluation.”

In Figure 1 we depict the structure and content of the SQaRE series. Each division of the SQaRE series is intended to support software quality evaluation.

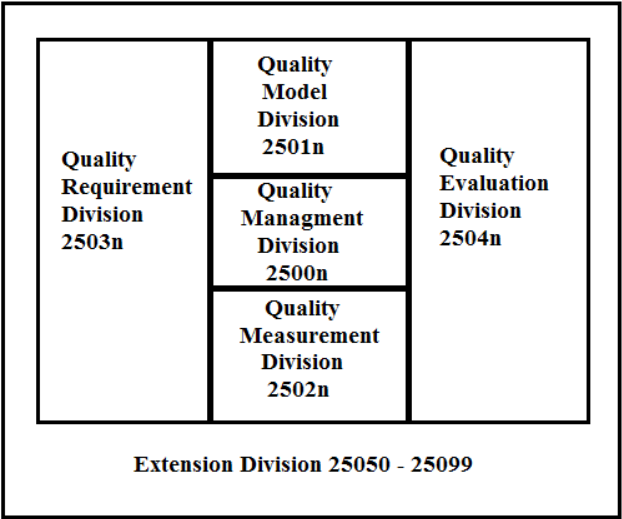


Figure 1. The SQaRe series of standards: [2]

This proposal is focused on the third division, which includes ISO/IEC 25023 standard. ISO/IEC 25023 contains external and internal quality measures for evaluating the system/software product quality [3]. Both external and internal measures can be used for quantitatively evaluating the software product when the development model used is iterative or incremental. The quality measures defined in this standard are intended to be implemented in terms of the next characteristics: functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, portability [3].

## 2.2 Software engineering method and theory (SEMAT)

SEMAT is an initiative proposed to address some of the major problems the software engineering faces today [5]. SEMAT initiative includes a set of widely-agreed elements and solid theoretical basis of the software engineering, which can be used for practice composition, enactment of methods, and use and improvement of methods. SEMAT is focused on three areas of concern, represented by colors: Customer (green), Solution (yellow), and endeavor (blue), as shown in Figure 1. The first area is focused on the actual use of the software system, the second area is focused on specification and development of the software system, and the third area is focused on the team and their way of working [5].

The SEMAT kernel includes essential elements to work with in every software development endeavor (the so-called alphas): opportunity, stakeholder, requirements, software system, work, team, and way of working. They can be used for assessing the health and progress of a software engineering endeavor [7]. Descriptions of what a team commonly produces and uses during the development of a software product—like requirements and design—can be provided by the alphas [5]. Alphas are different from work products, since alphas represent critical indicators of the progress and health of a software product. Each alpha includes a set of states, which have associated checklists. In Figure 2 we show the alphas of the SEMAT kernel.

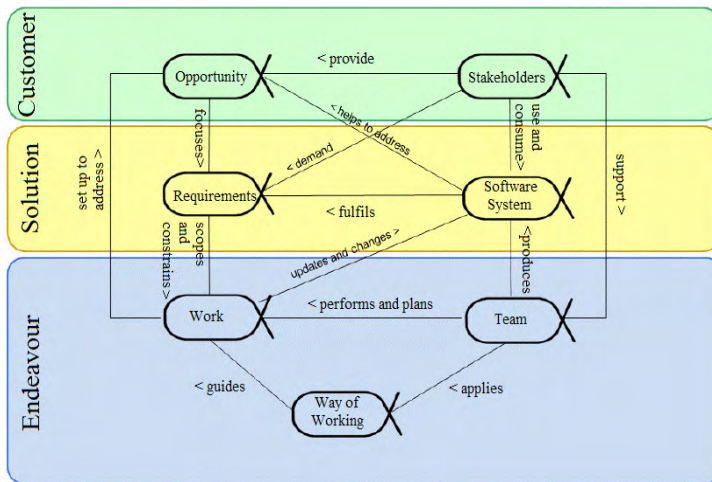


Figure 2. Alphas of the SEMAT kernel. Source: [2]

## 2.3 ISO/IEC 9126 standard overall view

ISO/IEC 9126 standard comprises four technical reports: 1) Quality model; 2) External metrics; 3) Internal metrics; and 4) Quality in use metrics. The quality model includes two parts for software product quality. First part is an introduction of external and internal quality, which is specified in detail in Part 2 and Part 3 of the ISO/IEC 9126 standard. Second part of the quality model is an introduction to quality-in-use metrics. External metrics are implemented during the testing stages of the software development life cycle for measuring the behavior of the system [9]. “Internal metrics are applied to the non-executable software product” and they are useful to identify quality issues in the early stages of the software development life cycle [10]. Finally, quality-in-use metrics are implemented for measuring the way a software product meets the needs of users in order to achieve specified goals [11]. Parts 2 and 3 of the ISO/IEC 9126 standard

are related to six characteristics—functionality, reliability, usability, efficiency, maintainability, and portability—, which are further subdivided into sub-characteristics. Part 4 is related to four characteristics—effectiveness, productivity, safety, and satisfaction—with no sub-characteristics [8, 11].

In Figure 3, we show the influences among the three types of metrics. So, internal quality influences the external quality and quality in use is highly influenced by the external quality. Quality in use depends on external quality, and external quality depends on internal quality, *i.e.* quality begins from the very early stages of the development of the software product. ISO/IEC 9126 standard was developed in order to assess software product quality by using metrics, which can be modified by the users of the standard [8]. Each quality characteristic has associated sub-characteristics and metrics.

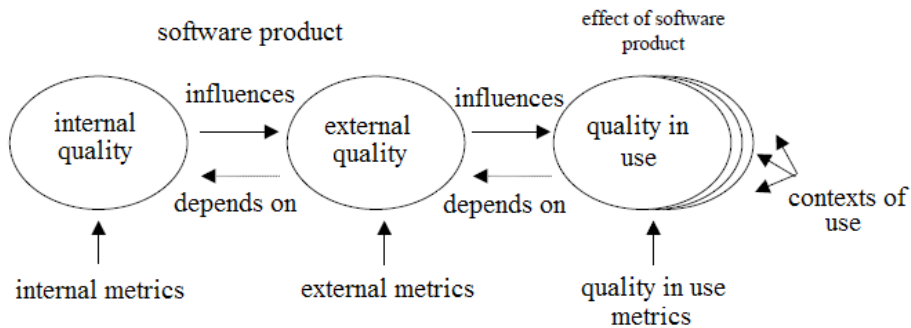


Figure 3. Relationship between types of metrics. Source: [8]

### 3. State-of-the-art review

#### 3.1 Associating performance measures with perceived end user performance: ISO 25023 compliant low level derived measures

Ravanello *et al.* [12] propose an improvement of the service level agreements for cloud computing applications. Measures of the ISO/IEC 25010 standard are applied in order to predict the degraded state of a private cloud computing application. The relationship between the ISO/IEC 25010 measures and low level derived measures is underspecified in this work. They implement measures associated with the efficiency characteristic. Normal, abnormal, adequate, and degraded are the indicators they implement for modeling the state of a large private computing application.

### 3.2 A new quality in use model for mobile user interfaces

Alnanih *et al.* [13] propose a new quality-in-use model for measuring user interface design quality. This model is based on the ISO/IEC 9126-4 technical report and it is intended to be specifically used in mobile devices. The proposed model includes a new factor or sub-characteristic according to the ISO/IEC 9126 called “Task Navigation” and aims at helping the designer to evaluate the mobile application while executing work-related tasks. This proposal is intended to be used for mobile devices instead of any other software product. Some inconsistencies related to the terminology the ISO/IEC 9126 standard arise, since they use different terminology for term characteristic and metric.

### 3.3 Tutorial about SEMAT initiative and MetricC game

Zapata *et al.* [14] present a tutorial about a SEMAT-based game called MetricC. The game allows players for understanding activity spaces, which are defined in the SEMAT kernel, as well as completion criteria and metrics. Players need no previous knowledge about SEMAT. They use some ISO/IEC 9126 metrics for measuring checklists of the alpha states. Relationship between alpha states and ISO/IEC 9126 metrics is subjectively defined by the game authors. This tutorial has a brief specification of which metrics of the standard they are using, which leads to misunderstand the metrics they relate to the activity spaces.

### 3.4 Enhancing ISO/IEC 25021 quality measure elements for wider application within ISO 25000

St-Louis and Suryan [4] propose an approach for determining a set of quality measure elements as a way to improve the applicability of ISO/IEC 25010. They define a core set of measures excluded from the ISO/IEC 25010 measures. This proposal is aimed to define “a precise set of base quality measures, which would serve as a basis for further development of the majority of derived quality measures.” [4] The authors claim the core set of measures is easier to understand and apply.

### 3.5 On the relationship of concern metrics and requirements maintainability

Conejero *et al.* [15] introduce an empirical analysis of the correlation among crosscutting properties and two ISO/IEC 9126 maintainability attributes, namely changeability and stability. The chapter has some inconsistencies related to the terminology the authors are using to describe the ISO 9126 standard; they use the words *quality attri-*

*butes and sub-attributes* instead of *characteristics and sub-characteristics* which are the right words to use when describing the ISO/IEC 9126 quality model. The authors are focused on answering the next question: *How do scattering, tangling and crosscutting affect requirements maintainability?* The authors correlate some concern metrics with maintainability and they exclude other characteristics.

### 3.6 Metric proposal for system testing models verification of safety critical systems

Spendla *et al.* [16] propose a new metric for verifying the system testing model for critical systems. The proposed metric is based on the ISO/IEC 9126 standard and it is intended to verify communication subsystem—a specific area of critical systems. The value of this metric determines whether the system testing model meets the specified requirements or not. This metric is based on the *adequacy* metric of the ISO/IEC 9126 standard. This proposal can be only used for newly designed system testing models of safety critical systems, which make the proposal so limited.

### 3.7 Customizing ISO 9126 quality model for evaluation of B2B applications

Behkamal [17] proposes a quality model for evaluating B2B applications by customizing the ISO/IEC 9126 quality model in order to identify acceptance criteria and evaluate a particular application domain. The quality model has five steps: choosing ISO quality model as a basis; Identifying quality characteristics of B2B applications; choosing a group of software experts; assigning weights to the quality factors and sub-factors and developing quality criteria. The customization of the ISO/IEC 9126 model is done by extracting the quality factors from web applications and B2B e-commerce applications, weighting these factors from the viewpoints of both developers and end users, and adding them to the new model. This chapter exhibits some inconsistencies related to the terminology the authors are using for describing the ISO/IEC 9126 quality model. For example, they use the words *quality factors* instead of *characteristics*. The proposal is restricted to B2B applications. Also, metrics for implementing B2B applications are subjectively defined by the authors.

### 3.8 On the relationship of the ISO/IEC 9126 metrics and the alpha states of the SEMAT kernel

Zapata and Montoya [18] propose some ISO/IEC 9126 metrics to be related to the requirements and the software system alpha states. The ISO/IEC 9126 standard was re-

placed by the SQuaRE standard, then the metrics should be replaced by ISO/IEC 25000 measures.

## 4. The ISO/IEC 25023 measures and alpha states relationship

In this Section, we explain each alpha and then we propose the relationship between ISO/IEC 25023 measures and the alpha states. We use the terms *metric* and *measure* in our proposal to refer to “a measurement scale and the method used for measurement” [8]. The term *metric* is used in ISO/IEC 9126, while the term *measure* is used in ISO/IEC 25000—SQuaRE. Their meanings are basically the same.

The SEMAT kernel alphas are used to address three issues: 1) they capture the key concepts of software engineering; 2) they support the assessment of health and progress of a software endeavor; and 3) they are used to define software engineering methods and practices [5]. Each one of the SEMAT kernel alphas has a set of states. Alpha states are implemented when assessing the health and progress of a software endeavor [18]. Each alpha state has associated a checklist. A state can be only achieved when the entire checklist items are met [5]. “There is the possibility of iterate through the states if necessary, it depends on your choice of practices.” [5]

According to the ISO/IEC 25023 standard, both external and internal measures can be used for quantitatively evaluating the software product when the development model used is either iterative or incremental [3]. Internal metrics play an important role in the development process, since they can be used to predict the quality of the final product [10]. System/software specification, architectural design, detailed design, component and code can be measured by using internal metrics [3]. External metrics should be applied to executable software and they can be only used during testing stages of the software development life cycle [9]. In our study, only two alphas can be related to ISO/IEC 25023 measures: requirements and software system [18]. The remaining alphas have no relation with the ISO/IEC 25023 standard, since such standard is focused on software product quality by evaluating eight characteristics related to technical factors—functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, portability [3]. Consequently, customer and endeavor—the remaining areas of concern—are uncovered by the standard.

## 4.1 Requirements

“Requirements are what the software must do to address the opportunity and satisfy the stakeholders” [5]. Requirements capture the needs of the stakeholder to solve a problem or to achieve a specific goal [19]. During the development of a software product, the requirements are reprioritized and adjusted to the needs of the stakeholders [5]. “This allows the requirements to act as a verifiable specification for the software system.” [5] Requirements can be verified and evaluated during the testing stages of the software development process. SEMAT includes six states to track the health and progress of the requirements, as shown in Figure 4. Requirements can change during the software development life cycle until they are acceptable for the stakeholder, but “it is essential that they stay within the bounds of the original concept and that they remain coherent at all times” [5].

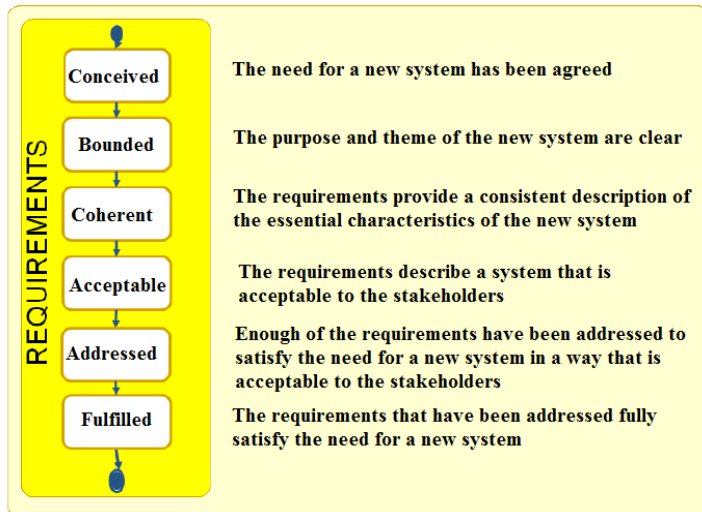


Figure 4. The states of the requirements. Source: [5]

## 4.2 Software system

“A software system is made up of software, hardware, and data that provides its primary value by the execution of the software.” [5] During the development of a software system, six states can be identified according to the SEMAT kernel: architecture selected, demonstrable, usable, ready, operational, and retired. In Figure 5, we show the states of the software system [18]. Each state provides stability during the software system development process.



Our proposal is focused on the selection of the appropriate ISO/IEC 25023 measures to evaluate and validate the chosen states of the alphas—in this case requirements and software system. The relation is based on similar terminology and meanings of the ISO/IEC 25023 measures and the checklist of the chosen alphas belonging to the SEMAT kernel [18].

We first analyzed the alpha *Requirements* and its states. We chose the state *Addressed* to be measured by ISO/IEC 25023 measures. The selection of the state is mainly based on its checklist, as shown in Table 1. The checklist includes four items and supports the evaluation of health and progress of the requirements. All of the items should be met in order to obtain the state. “In the addressed state the amount of requirements that have been addressed is sufficient for the resulting system to provide clear value to the stakeholders” [5].

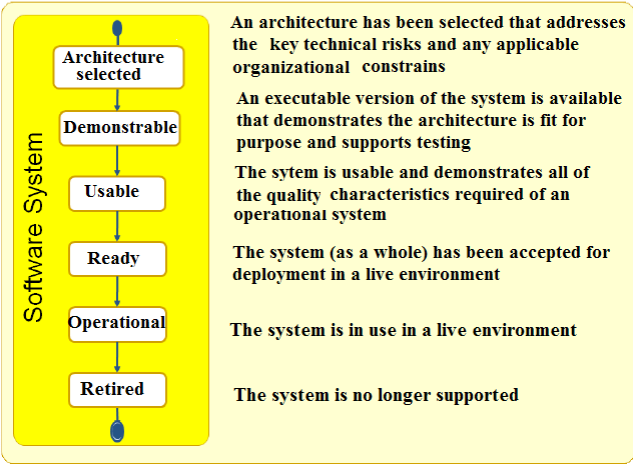


Figure 5. The states of the software system. Source: [5]

Table 1. Addressed state of the alpha requirements

Checklist	
Alpha state	Items
Addressed	Enough of the requirements are addressed for the resulting system to be acceptable to the stakeholders.
	The stakeholders accept the requirements as accurately reflecting what the system does and does not do.
	The set of requirement items implemented provide clear value to the stakeholders.
	The system implementing the requirements is accepted by the stakeholders as worth making operational.

We identified *functional coverage*—what portion of the specified functions has been implemented— as a direct measure to verify and validate the *addressed state* [3]. This measure is part of the set of measures of the functional completeness sub-characteristic, and this sub-characteristic is part of the functional suitability characteristic. The relevant measures identified are: *functional coverage*, *functional correctness*, *functional appropriateness of usage objective*, and *functional appropriateness of system*. The relevant measures identified can all be applied for evaluating the state *addressed*. There are more measures that can be applied for validating this state but for the scope of this Chapter, we only selected one measure: *functional coverage*.

$$\chi = 1 - (\alpha/\beta) \tag{1}$$

$\alpha$  = Number of functions missing.

$\beta$  = Number of functions specified.

ISO/IEC 25023 standard “does not assign ranges of values of the measures to rated levels or to grades of compliance because these values are defined based on the nature of the system, product, or a part of the product” [3].

The chosen measure is adequate for validating the functions specified in the requirements, since the missing functions are counted and compared with the requirements specification, which can lead to an acceptable set of requirements for the stakeholder as worth making operational, which is required by the addressed state. This metric helps to obtain quality requirements and achieve the state *addressed* by helping to identify how many functions are missing in terms of the specified functions.

Finally, we analyzed the alpha *Software System* and its states. The state *operational* is chosen. The selection of the state is based on its checklist as well as for the alpha *Requirements*. The checklist has three items, as shown in Table 2. The three items have to be met in order to obtain the state.

Table 2. Operational state of the alpha software system

Checklist	
Alpha state	Items
Operational	The system has been made available to the stakeholders intended to use it.
	At least one example of the system is fully operational.
	The system is fully supported by the agreed service levels.

The process to select the measure for the *operational (the system in use in a live environment)* state is the same we implemented to select the measure *functional coverage* of the alpha *Requirements*. We identify the characteristic *usability* as a direct measure to validate and verify the state. *Usability* has six sub-characteristics: *appropriateness* *recognizability*, *learnability*, *operability*, *user error protection*, *user interface aesthetics* and *accessibility* [3]. We selected the sub-characteristic *operability* to verify and validate the state *operational*; such sub-characteristic contains nine measures. The measure we chose for measuring the *operational* state is *operational consistency in use*. This measure determines “to what extent do interactive tasks have a behavior and appearance that is consistent both within the task and across similar tasks” [3]. The measure equation is:

$$c = 1 - (\alpha/\beta) \quad (2)$$

$\alpha$  = Number of specific interactive tasks that are performed inconsistently.

$\beta$  = Number of specific interactive tasks that need to be consistent.

As we mention before, the ISO/IEC 25023 standard does not assign ranges of values of the measures.

Several operation patterns are recognized by means of user experience [9]. The chosen measure is adequate for validating the consistency of interactive tasks and other tasks and the way they are performed. The software system can be considered operational based on the consistency of the task performance, since one of the checklist items requires at least a fully operational example of the software system.

According to the ISO/IEC 9126 standard “metric is defined as a measurement scale and the method used for measurement,” which are used for obtaining indicators. An indicator is a metric or a combination of metrics that provides an important view of the software product and helps to make better decisions during the software development life cycle [20].

## 5. Conclusions

In this Chapter we explored the state of the art; presented a theoretical framework which contains an introduction about ISO/IEC 25000 standard, ISO/IEC 9126 standard, and The SEMAT kernel; presented new lines of research; and finally presented an analysis of the measures of the ISO/IEC standard to determine which of them can be relate to the alpha states of the SEMAT kernel. Concerning the state of the art, we have presented a rela-

tionship between two alpha states—addressed requirements and operational software system—and the ISO/IEC 25023 measures.

Measuring the alpha states by using the ISO/IEC 25000 measures guarantee a higher level of quality requirements and software systems based on the validity and acceptability of this measures for verifying the quality of a software product.

## 6. Future work

SEMAT initiative includes the aforementioned seven alphas. We only presented two of the seven alphas and analyzed the ISO/IEC 25023 measures for associating the alpha states. As future work, we propose the exploration of other standards in order to evaluate and verify the states of the five remaining alphas, *opportunity*, *stakeholder*, *work*, *team*, and *way of working*. In this chapter, we only focused on one state of the alpha *requirement* and one state of the alpha *software system*; we propose another line of research to relate the ISO/IEC 25023 measures to every state of both alphas in order to verify and validate them during testing stages of the software development.

## 7. References

- [1] A. Abran, R. Al-Qutaish “ISO 9126: Analysis of Quality Models and Measures,” In: Software Metrics and Software Metrology, A. Abran (Ed.), Wiley-IEEE Computer Society, New York, USA, pp. 205–228 (2010).
- [2] Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE), International Organization for Standardization, ISO/IEC 25000, 2014.
- [3] Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – Measurement of system and software product quality, International Organization for Standardization, ISO/IEC 25023, 2016.
- [4] D. St-Louis, W. Suryn, “Enhancing ISO/IEC 25021 quality measure elements for wider application within ISO 25000 series,” in IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society, Canada, 2012, pp. 3120–3125.
- [5] Essence – Kernel and Language for software Engineering Methods, Object Management Group. 2015.
- [6] C. Zapata, and I. Jacobson, “A first course in software engineering methods and theory”, DYNA vol. 81, pp. 1–11, February 2014.
- [7] I. Jacobson, P. Ng, P. McMahon, I. Spence, and S. Lidman, The Essence of Software Engineering: Applying the Semat kernel, 1st ed, Boston, Pearson education: United States of America, 2013.

- [8] Information Technology - Software Product Quality - Product Quality- Part 1: Quality Model, International Organization for Standardization, ISO/IEC 9126-1, 2008.
- [9] Software Engineering - Product Quality - Part 2: External Metrics, ISO/IEC 9126-2, 2000.
- [10] Software Engineering - Product Quality - Part 3: Internal Metrics, ISO/IEC 9126-3, 2000.
- [11] Software Engineering - Software Product Quality - Part 4: Quality in use Metrics, ISO/IEC 9126-4, 2000.
- [12] A. Ravello, L. Villalpando, J. Desharnais, A. April, and A. Gherbi, "Associating performance measures with perceived end user performance: ISO 25023 compliant low level derived measures" in *Cloud computing 2015*, France, 2015, pp. 120-125.
- [13] R. Alnanih, O. Ormandjieva, and T. Radhakrishnan, "A New Quality-in-Use Model for Mobile User Interfaces," in *2013 Joint Conference of the 23rd International Workshop on Software Measurement (IWSM) and the Eighth International Conference on Software Process and Product Measurement (MENSURA)*, 2013. pp 165-170.
- [14] Zapata, J.C., Vianney, M.G., and Castro, L.F., "Tutorial sobre la iniciativa Semat y el juego MetriCC," in *Congreso Colombiano de Computación*, Colombia., 2013, pp. 1-3.
- [15] J. M. Conejero, E. Figueiredo, Garcia, A. J. Hernández, and E. Jurado, "On the relationship of concern metrics and requirements maintainability," *Information and Software Technology* vol. 54, pp. 212-238, February 2012.
- [16] L. Spendla, p. Tanuska, and I. Smolarik, "Metric Proposal for System Testing Models Verification for Safety Critical Systems," in *IEEE 11th International Symposium on Intelligent Systems and Informatics*, Serbia, 2013, pp. 87-92.
- [17] B. Behkamal, M. Kahani, and M. K. Akbari, "Customizing ISO 9126 quality model for evaluation of B2B applications," *Information and Software Technology* vol. 51, pp. 599-609, March 2009.
- [18] C. Zapata, Y. Montoya, "On the relationship of the ISO/IEC 9126 metrics and the alpha states of the SEMAT kernel" in *2016 4th International Conference in Software Engineering Research and Innovation (CONISOFT)*, Mexico, 2016, pp. 59-64.
- [19] M. Saadatmand, A. Cicchetti, M. Sjodin, "Toward model-based Trade-off analysis of non-Functional requirements," in *38th Euromicro Conference on Software Engineering and Advanced Applications*, Sweden, 2012, pp. 142-149.
- [20] R. S. Pressman, *Ingeniería del Software un enfoque práctico*, 5th ed, C. F. Madrid, Mc Graw Hill: España, 2002.

# Chapter # 4

## Computational tool for a communication system for persons with tetraplegia using an eye-tracking sensor

Oscar Abraham Grijalva Hernández, María Trinidad Serna Encinas,  
César Enrique Rose Gómez, Oscar Mario Rodríguez-Elías  
Instituto Tecnológico de Hermosillo, División de Estudios de Posgrado e Investigación,  
Maestría en Ciencias de la Computación  
oagrijalva@gmail.com, tserna@ith.mx, crose@ith.mx, omrodriguez@ith.mx

### 1. Introduction

Nowadays, approximately 5% of Mexico's population suffers from some type of impairment, according to the National Institute of Statistics, Geography and Informatics (INEGI, for its acronym in Spanish), which represents more than 5,700,000 people of all ages [1]. The motor impairment is affecting more people, and can be caused by diseases, accidents, birth defects, and by the elderly, among others.

Tetraplegia is a type of physical impairment; it results in impairment of function in the arms as well as typically in the trunk, legs and pelvic organs, including the four extremities [2]. Tetraplegia develops when the spinal cord is injured, preventing the communication between the central nervous system and other organs and body structures [3].

The inability to express and communicate their basic needs affects the emotional state of the person with the impairment. A person unable to express or communicate their desires and needs are prone to depression, a condition that causes sadness and hopelessness [4].

Quadriplegia affects in a negative way the life quality, as well as the emotional status of the people that have this condition. To bring the opportunity of communication to the patients with this disease, is the reason for the implementation of a system for the communication using an eye-tracking sensor, so they can express in a written and oral way.

Therefore, the challenge was to solve the next problem: What functional factors must be considered to design a friendly and intuitive graphical interface, such that the patient with quadriplegia can have interaction and communication with relatives and medical staff in a written and orally way?

In order to provide the ability to communicate to patients with this impairment, in this paper we propose the implementation of a computational tool with a graphical interface, through the use of an eye-tracking sensor, that allows the communication of quadriplegic patients using oral and written expressions.

This paper presents the design and development of the computational tool described above, for which, in the next section the theoretical foundations that support the proposal are presented, then, in the section 3, the methodology used to obtain the requirements for the analysis and design of the graphical interface is showed, after that, in sections 4 and 5 the development and the results are shown respectively, and finally, our conclusions are set in section 6.

## 2. Theoretical Framework

This paper focuses on the design of a graphical interface to users with specials needs, particularly patients with tetraplegia. In this sense, among the issues that are important to understand, there are the impairment and impairments types, later is discussed the tetraplegia as motor impairment. Also, it is important to identify the mechanisms by which information may be provided by a quadriplegic patient, such as sensors, and particularly the eye-tracking sensor that was chosen for the proposed system.

### 2.1 Impairment

Impairment is a term to define the problems that affect or diminish the functions or body structures of a person, as anomalies or defects in the body's organs, or the loss of such organs. Impairment limits the participation of the person who has it in daily activities, due to the difficulties to perform different actions [5, 6].

Different types of impairment are classified as follows [7]:

- Sensory and communication impairment. Such as visual and hearing impairment. Also, those that are affecting or impeding the communication and language comprehension.

- » Visual impairments. These include the complete or partial loss of vision and other impairments that cannot be improved with the use of visual aids.
- » Hearing impairments. These consist of the loss of hearing in one or both ears.
- » Communication and language comprehension impairments. These implicate the impairments to express and natural language understanding.
- Motor impairments. These impairments affect the functioning of the upper extremities, lower extremities, trunk, head and neck.
  - » Lower extremities impairments. These are comprised of loss of movement of the legs and related body structures.
  - » Upper extremities impairments. These consist of complete or partial loss of movement of the arms.
- Mental impairments. Include the deficiencies in mental status, loss of memory, and behavioral problems of the individual.
  - » Intellectual impairments. These deal with lower than average intellectual capacities.
  - » Behavioral impairments. These include the changes in the behavior of those who suffer them.
- Multiple impairments. Are the cases in which there are two or more impairments, hence, we talk about of multiple impairments, for instance, a cerebrovascular accident that affects motor and intellectual capacities of a patient.

## 2.2 Tetraplegia

This term refers to impairment or loss of motor and/or sensory function in the cervical segments of the spinal cord due to damage of neural elements within the spinal canal. Tetraplegia results in impairment of function in the arms as well as typically in the trunk, legs and pelvic organs, depending on the extension of the damage, the breathing and speaking capabilities can also be affected. [2, 8].

An injury to the spinal cord can be caused by an accident, a disease, a tumor, an electric shock, poisoning or lack of oxygen, and it is not required that the spine is fully affected to cause tetraplegia. When the spinal cord gets damaged, the nervous system finds itself incapable of communicating signals to the parts of the body connected to the vertebrae



below the place of the injury; to cause tetraplegia, the upper section of the spine, known as the cervical spine, must get damaged (see Figure 1).

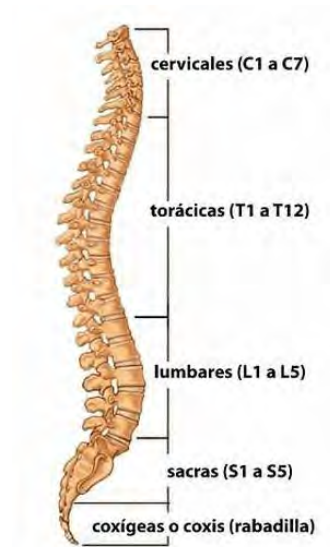


Figure 1. Spine

Cervical 1-4: These are the first cervical of the spinal cord, such that the risk is high to get impaired people. Patients with injuries at C3 cervical or higher may be unable to breathe on their own, and they can need a respirator. Patients with lesions at C4 cervical can be free of respiratory problems, but sometimes needs a respirator. These people can lost movements and sensation in the extremities, its common that they have movements in shoulders and neck.

Cervical 5: Patients with C5 cervical lesions have deltoid muscle and functional biceps. They can slightly rotate the shoulders, hands and wrists but remain motionless. People with lesions in C5 can feed themselves and perform most daily functions.

Cervical 6: Patients with C6 cervical lesions are able to widely rotate their shoulders, bend their arms, but not extend them, and extend the wrists, and they can use the index finger and thumb. Persons with C6 injuries can usually drive adapted vehicles and take care of their own hygiene.

Cervical 7-8: Patients with lesions in cervical C7 and C8 have functional triceps, can bend and straighten their arms and they have more control over hands and wrists, these people can hold light objects, but with a limitation on his skill.

Lesions in the upper thoracic vertebrae (T1-T8), usually affect control of the upper torso, limiting its movement. Lesions in the lower thoracic vertebrae (T9-T12), allow good control of the torso and abdominal muscles. Lesions in the lumbar vertebrae (L1-L5) allow sending signals to the hip and legs of the patient. The sacral vertebrae (S1-S5), control signals to the groin, feet, and parts of the legs.

## 2.3 Ocular Sensors

A sensor is a device that allows us to perform a conversion of a signal from one physical form to another different physical form corresponding, so that the signal can be measured or processed [9].

The most common types of sensors that are applied to intelligent systems are as follows [10]:

- Acceleration. Sensors used to measure acceleration vectors of the objects in which they are used.
- Light. They are used to measure the light intensity at a specific point, commonly used to detect whether a device is within or outside a certain location.
- Proximity. They are used to determine the proximity of an object or person relative to the sensor.
- Audio. They are used to measure the sound signals.
- Temperature. These sensors are used to measure the temperature of a device or place.
- Movement. Motion sensors are useful for detecting movement patterns of objects or persons in order to analyze their behavior or activity.

In our work we are interested in eye-tracking sensors, because patients with symptoms of severe quadriplegia, as is particularly the case that has sought to address in this project, may have not control of most of their muscles, and only can communicate through eye movements.

The eye-tracking sensors measure movement done by one person with their eyes, allowing them to control electronic devices by means of sight, and achieving interaction without the need for hands. The “Eye Tribe Tracker” sensor was chosen for use with the

communication system for quadriplegics, this sensor monitors the pupils of the users, identifying X and Y coordinates for each of their eyes, and calculating the specific point of sight (see Figure 2) [11].

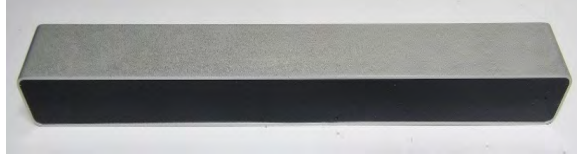


Figure 2. Eye Tribe Tracker sensor

## 2.4 Related work

There exist several information systems developed for helping people to communicate by means of proposals in the fields of augmentative and alternative communication. An example is GESTELE, a system that allows users to communicate their ideas and moods, using a virtual keyboard, and a set of icons which represent emotions [12]. Another example is GazeSpeaker, a system focused on helping people with communicative disabilities. GazeSpeaker also uses a graphical user interface to help users to choose an option from a set of preselected phrases. This system can be also used with eye-tracking sensors, such as Eye Tribe Tracker [13].

Although GESTELE uses a graphical interface to operate, it requires the user's physical interaction to choose options, therefore it has limitations that hinder its use with patients with severe motor impairments. On the other hand, Gazespeaker does not provide functionalities to facilitate its adaptation to patients with severe tetraplegia whom eye movements are partially compromised, as is our case. Additionally, both systems don't considers the use of alerts related to the needs of the patient.

## 3. Basic Functional Requirements

The main part of the work focuses on identifying the requirements of interaction between the patient and the system. To determine these requirements, we have the support of a patient presenting advanced tetraplegia, this patient does not have movement in her upper and lower extremities as well as it is limited in her ability to speech. The patient has only eye movement, and mental and hearing abilities are normal. Due to the particular situation of the patient, her interaction with family members, in particular how to communicate was observed. This allowed defining the functional requirements for the graphical interface.

Communication between the patient and her family or carer is done through a series of options grouped by categories that represent physical issues, which are presented to the patient orally, and which she selects the desired using the blinking of her eyes. According to the selected category, corresponding subcategories are presented, and the patient chooses one by blinking; the process continues until the patient has selected a specific concept, for example, headache.

**Functional requirement 1:** According to the observations previously mentioned, it was established as a basic requirement that the GUI system to be developed should present content organized by categories, in order to facilitate the selection of deployed options during patient interaction.

Through the interaction with the patient, we also found that the patient could only perform eye movement in ascending or in descending order. The gaze of the patient is shown in Figure 3.

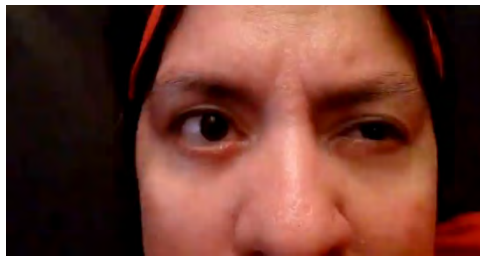


Figure 3. Patient gaze

Figure 3 shows the gaze of patient in a centered image, it can be seen how the patient has difficulty for holding her left eyelid, a situation that causes that the readings of this eye, obtained by the sensor eye-tracking, be unacceptable for usage in the graphical interface. Therefore, as a functional requirement 2, only readings generated from the movement of the right eye will be used. Figure 4 shows the patient eye movement with upward direction, while Figure 5 shows the patient looks downward, where it can be seen how the patient's left eye is completely covered by the eyelid when looking downward, causing these readings to be also unacceptable for use in the GUI.

During the interaction with the patient, it was found that there are cases in which the patient seeks to communicate a need not preset to her families; in these cases, it is presented to the patient two sets of letters to choose from, consonants and vowels, the patient selects the desired group by blinking her eyes. If vowels are selected, her relatives present to the patient each vocal individually, until one is selected. If consonants are

selected, the patient is asked to choose between two groups of letters; the first from B to M, and the second one from N to Z, by selecting the group, the corresponding letters are presented individually until one is selected. By selecting any of the previously submitted letters, the relative or carer turns to perform the process to form a complete word. The process is repeated for each word, so at the end, a phrase is obtained.



Figure 4. Patient looks up

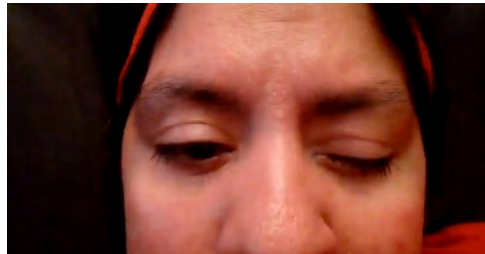


Figure 5. Patient looks down

By analyzing the patient's interaction with her caregiver, previously described, the functional requirement 3 was acquired, which provides the patient with a mechanism for choosing letters that allow her to form words and sentences to communicate non-existent concepts in the categorized options.

Once identified these three basic functional requirements, we proceeded with the implementation of the system, described below [14].

## 4. Development

To solve the problem discussed above, and following the three basic functional requirements identified and described previously, a communication system for quadriplegics is proposed. This system uses a graphical user interface, which allows the patient to set her sights on the graphic components displayed in the interface and through the

eye-tracking sensor transform the gaze in coordinates representing specific points on that interface. These points are used to perform certain actions, for instance, to enter a category or display a phrase, and allow the patients to communicate their needs or requirements to their caregivers.

## 4.1 System architecture

The architecture designed for the communication system, shown in Figure 6, consists of three modules: data input module, processing module and data output module. The first two modules have been developed following the obtained functional requirements, and the last one is being developed. The first two modules, the data input module and the processing module directly affect the interaction between the patient and the graphical interface presented in the communication system.

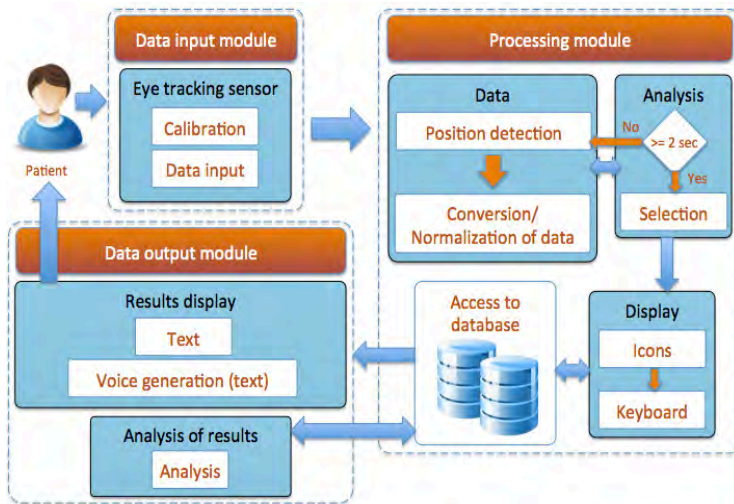


Figure 6. System architecture

**Data input module:** This module shows that reading of the data will be made by means of a sensor, which must be calibrated, and subsequently used to record eye movements, sending this data to the processing module.

**Processing module:** This module is divided into three processes, which are described below:

**The data registration process:** Focuses on the detection of the exact position that the user is being focused on the screen.

The analysis process: This process make data analysis of the registration process, if the condition is true when it is fulfilled, then the corresponding selection is executed.

The process of display: The display process is responsible for displaying on the screen the elements corresponding to the user selection, as in a menu change, which can be consulted in the database.

Data output module: This module is responsible for displaying the results on the screen, and if it's necessary play the corresponding audio.

## 4.2 Analysis and design

In Figure 7 the context diagram for the communication system is shown.

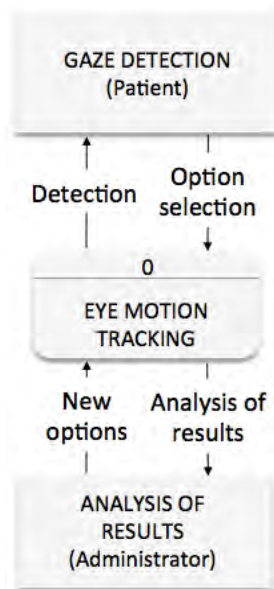


Figure 7. Context diagram: Level 0 communication system.

The diagram has as main process the motion tracking. The process detects the user's gaze, and the patient can select the system options. Based on the option selected, an analysis of the results is generated, which can be used to add new options to the system.

In Figure 8 the higher level diagram for the communication system is shown.

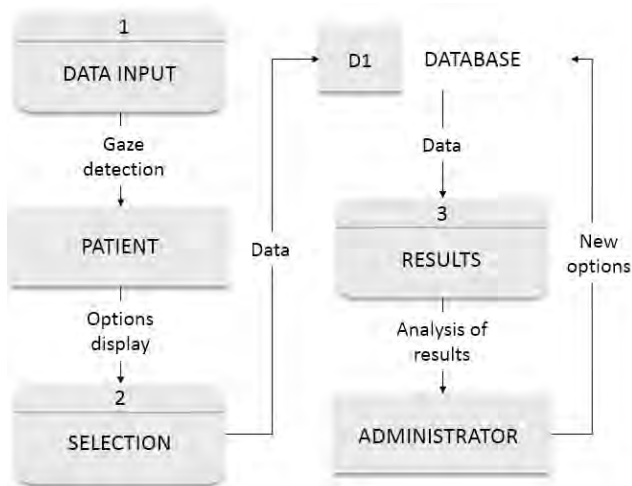


Figure 8. Higher level diagram: Level 1.

The previous diagram shows the system's three main processes: The data input process, the selection process and the process to generate results. The first process, the data input is responsible for detecting the patient's eye gaze. Then, in the selection process, the controls over which the user fixated his sight are determined and, if the selection is validated, its corresponding action is executed and the selection is stored in a database. Finally, in the results process, the data corresponding to the previous selections is shown on the graphical interface and, if it is needed, the data obtained are vocalized through a voice synthesizer, the analysis of results allows the system administrator add or modify content in the database.

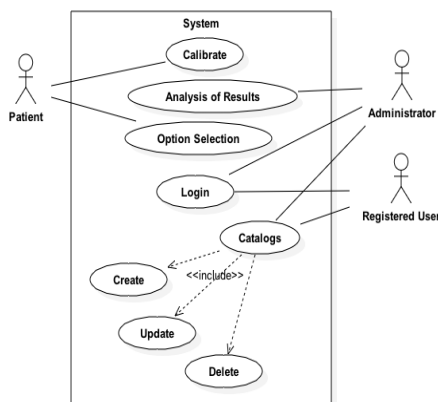


Figure 9. Use case diagram



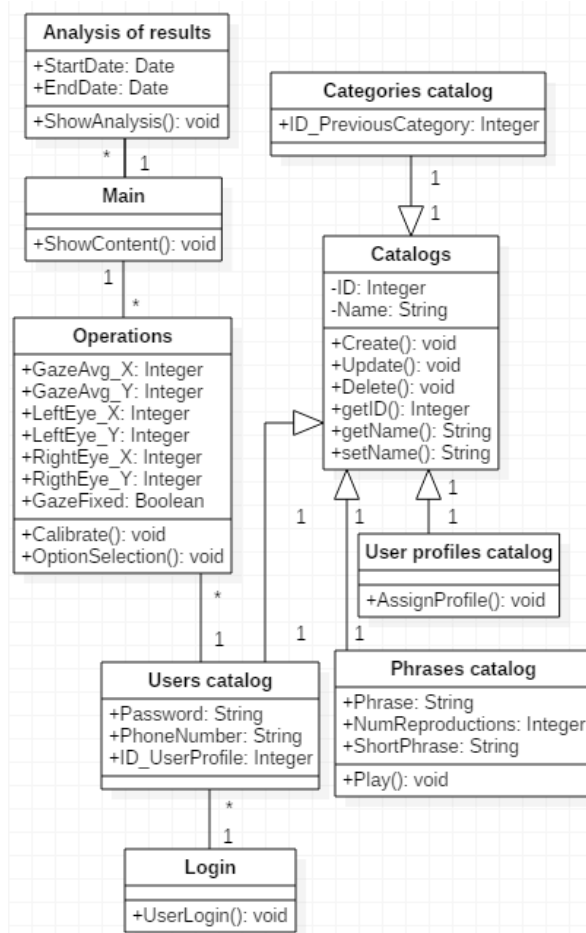


Figure 10. Class diagram

Figure 9 shows the use case diagram for the proposed system. There are three actors: Patient, Administrator and Registered User. The Patient is involved in the use cases: Calibrate and Option Selection. The Administrator is involved in the use cases for analyzing the results, Login, and Catalogs, which includes the functionality of use cases Create, Update and Delete. The Registered User can participate in use cases of Catalogs and Login.

Figure 10 shows the class diagram for the proposed system. The system's classes are: Catalogs, Users, User Profile, Categories, Phrases, Login, Operations, Main and Analysis of Results. The classes of Users, User Profile, Categories and Phrases inherit the func-

nality of the Catalogs class; the goal is to reuse the Create, Update, and Delete methods using the object-oriented design.

Login class allows logging of users. Operations class registers readings of the coordinates of the motion sensor and it is used to calibrate and select options on the screen. The Main class updates the content displayed on the screen. The Analysis of results class provides mechanisms to query the selected options within the system according to certain date parameters.

Activity diagrams are used to show the control flow and data flow within an operation performed by the system. The execution shown in an activity diagram can be sequential, concurrent, or both. These diagrams are composed of nodes of activity, which represent the execution of code statements within the system, and have connections to other nodes of activity. These activities are executed after the above has been completed.

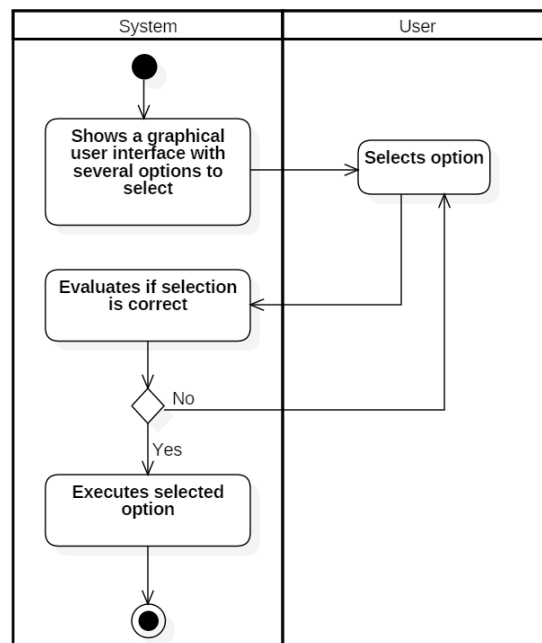


Figure 11. Activity diagram for selection of options

Figure 11 shows the activity diagram for the Selection of Options, in which the patient and the system are involved.

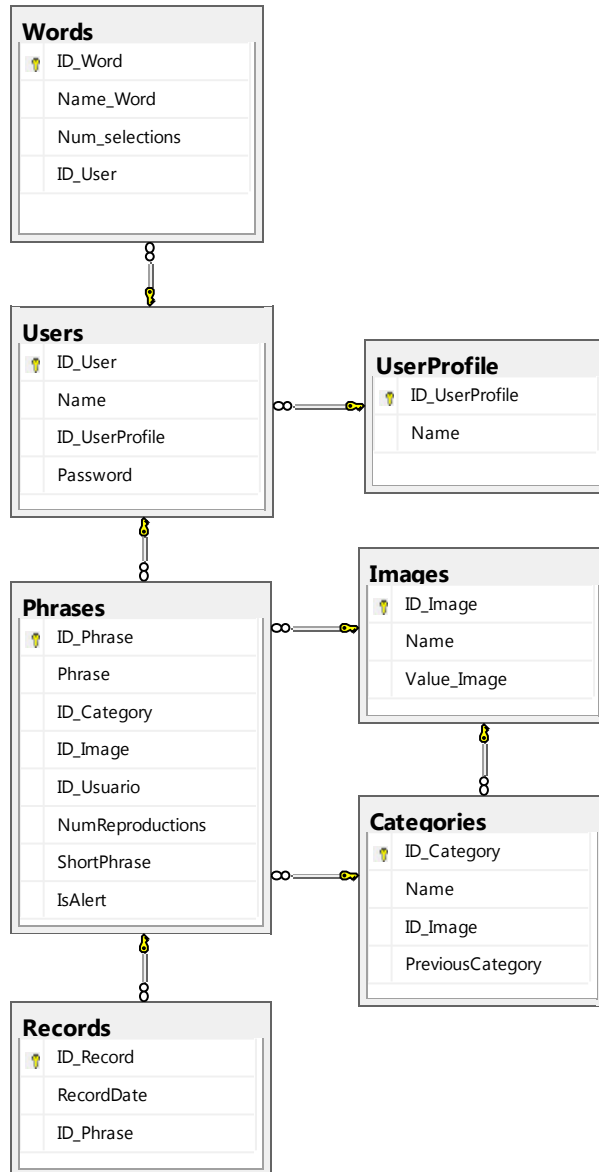


Figure 12. Database diagram

Figure 12 shows the diagram of the database created for the communication system, the database is composed of seven tables: the Users and UserProfile tables store information related to the system's users, as well as the available user profiles; the Categories, Phrases and Images tables store information about the preset phrases in the system, as

well as its hierarchy of categories and subcategories; the Words table stores the words used in the system and their frequency of use; the Records table contains the records of phrases selected in the system.

### 4.3 Communication system

In this section the implementation of the communication system is described.

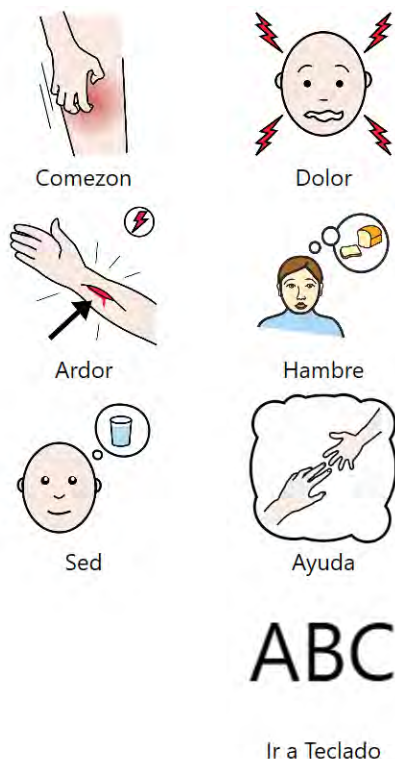


Figure 13. Initial categories

Figure 13 shows the categories for the initial settings of the GUI. These options are displayed in two columns vertically, due to the patient's inability to make a lateral movement with her eyes. The categories and subcategories of the interface use graphic components called pictograms, which allow the patient to associate a symbol with an established concept. These graphics have been developed by the Aragonese Portal of Augmentative and Alternative Communication (ARASAAC, for its acronym in Spanish), owned by the Provincial General of Aragon, and they're used under the Creative Commons License [15].

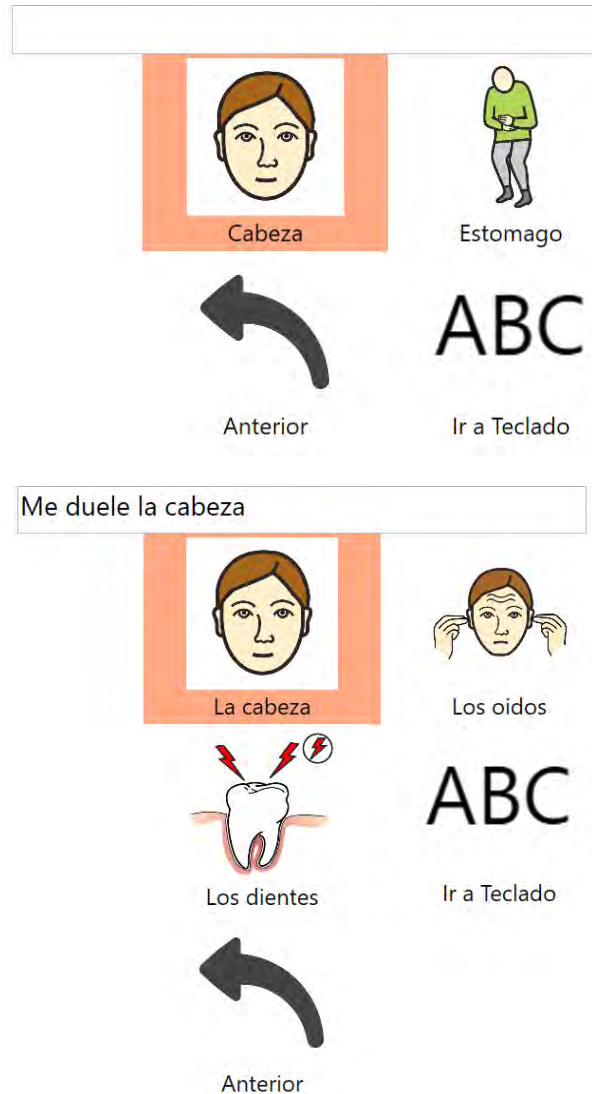


Figure 14. GUI by selecting a subcategory and a phrase

Figure 14 shows the process of selecting a phrase in the GUI of the communication system. To select a phrase, the patient must stare one of the options displayed on the screen. The eye-tracking sensor, in turn, recognizes the look of the patient and converts the position of the pupils to coordinates in the X and Y axes, representing a specific point on the interface. If this point is on a graphical component, it is highlighted changing its background color.

By selecting a category, the interface displays the inner categories as well as options to return to the previous level (with a direction arrow icon on the left), or access the keyboard (icon with ABC). When the patient agrees to the latter one subcategory, the interface displays the corresponding phrases options available (bottom columns); if the patient selects one of these icons, the GUI displays the text related to that option at the top, which is also reproduced by a speech synthesizer.

For example, selecting the category of pain, the interface displays the corresponding subcategories, as shown in Figure 14, when choosing subcategory of head, the interface displays the icons associated with it; by selecting the icon of the head, the interface has the phrase “Me duele la cabeza” (I have a headache) in the text field on the icons, also the phrase is played by the voice synthesizer.

```
private void MouseSeleccionaIconos(object sender)
{
    this.Dispatcher.Invoke((Action)(() =>
    {
        if ((sender as System.Windows.Controls.Button).Name
            .Equals("btnAtras"))
            botonAtrasClick();
        else if ((sender as System.Windows.Controls.Button).Name
            .Equals("btnFrasesaTeclado")){
            operaciones.Beep();
            botonFrasesaTecladoClick();
        }
        else
        {
            int x = Int32.Parse((sender as System.Windows.Controls.Button)
                .Name.Substring(3)) - 1;
            if (Datos[x, 3].Equals("1") || Datos[x, 3].Equals("2")){
                cantidadPalabrasSeleccionadas = 0;
                while (palabrasSeleccionadas==null)
                    palabrasSeleccionadas = Datos[x, 1].Split(' ');
                synth.SpeakAsync(Datos[x, 1]);
                IDFraseSeleccionada = Datos[x, 0];
                txtTexto.Text = Datos[x, 1];

                if (Configuraciones.smsGlobal)
                    EnviarMensaje(Datos[x, 1]);
                else if (Datos[x, 3].Equals("2"))
                    if (Configuraciones.smsHabilitado)
                        EnviarMensaje(Datos[x, 1]);
            }
            if (Datos[x, 3].Equals("0")){
                operaciones.Beep();
                miScroll2.ScrollToTop();
                ConsultaElementos(operaciones.InitializeDatabase(),
                    Int32.Parse(Datos[x, 0]));
            }
        }
    }
    ));
}
```

Figure 15. Code to select an icon on the graphical interface

Figure 15 shows a method used for the selection of icons on the graphical user interface. First, it is determined if the selected icon is an access button to a previous section or

to the GUI virtual keyboard, then, if the selected icon is a category or a phrase, its corresponding content is displayed on screen and, if it is a phrase, its content is vocalized through the voice synthesizer. If at least one alert condition is fulfilled, the content of the selected phrase is sent to the person taking care of the patient.

To meet the requirement where the patient with the help of her family or caregiver can form complex words and phrases not previously defined, it is incorporated into the graphical interface a keyboard that allows the patient to select individual letters in an orderly way, forming words and phrases. This keyboard is presented in Figure 16.

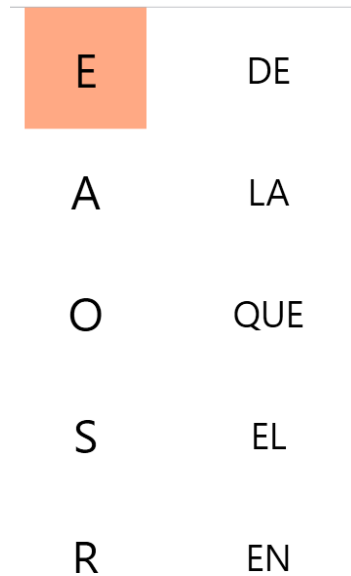


Figure 16. GUI keyboard

As shown in Figure 16, the GUI keyboard consists of two vertical columns. The first column, the column of letters, containing the 27 letters of the alphabet ordered based on their frequency of use in the Spanish language. The second column consists of common words suggested to the patient, according to the selected letter. When a new letter is selected in the first column, the interface displays new suggestions based on the letters selected by the patient, these suggested words are ordered based on their frequency of use within the system. Initially, the user interface considers the most common words in the Spanish language defined by the Royal Spanish Academy (RAE, for its acronym in Spanish) [16], in order to provide suggestions to the user when it begins to use the system. The words entered that are not in this catalog will be added to the system to be suggested later.

To facilitate the selection of letters on the screen keyboard, a special keyboard based in categories was also implemented, as it is shown in Figure 17.



Figure 17. Screen keyboard based in Categories

The categories use three groups of letters: consonants from letter B to letter M, consonants from letter N to letter Z and the vowels. On the second column, the most common words of the system are suggested to the user.

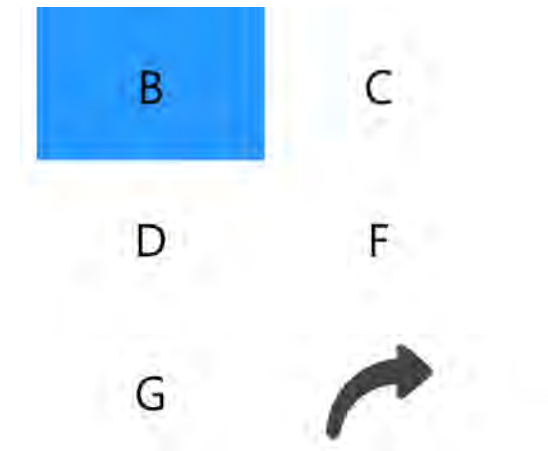


Figure 18. Categories based letter selection

On figure 18 the selection process of a letter contained in the first group of consonants is shown. The group's letters are divided into two parts, the first one, as it's shown on



the figure, goes from letter B to letter G, and the second one goes from letter H to letter M. The arrow pictured on the bottom of the right column allows the user to switch to the second group of letters in this category. Once a letter is selected, the graphical interface shows the most common words used on the system, suggested according to the user's current selection of letters.

## 5. Discussion

As it was shown on the previous section, the resulting graphical user interface complies with the three basic functional requirements identified to support the needs of a patient with severe tetraplegia. First, the incorporation of organized graphical icons adjusted to the patient's eye movement abilities. Then, the use of only the screen coordinates generated from the patient's right eye movement data. And finally, the addition of a mechanism that allows the patient to select letters, in order to form words and sentences to communicate needs and ideas not yet present in the categorized options. This mechanism is also organized by categories to facilitate the search and selection of letters, allowing the user to speed up the forming of words and sentences.

The graphical interface developed, unlike GESTELE, requires only eye movements to interact with the content shown on the screen of the system, allowing persons with severe motor impairment to communicate through it. Additionally, the proposed interface has been adapted to a patient who, additionally to her severe tetraplegia, her eye movements has limitations, therefore, unlike Gazespeaker, the patient does not require to have her ocular movement entirely functional to use the graphical interface. Additionally, we have also incorporated into the system, a module to send messages or alerts to the relatives of the patient. This allows the patient to communicate her necessities even when her relatives are not near, by using a wireless network or the cell telephone.

The computational tool of the communication system has been presented to the user (patient real) in order to perform some acceptance and usability tests. The patient has been able to use the interface to communicate with her relatives in an easy way. However, some difficulties have been observed, particularly, it has been difficult to the patient to accustom to the new mechanism. Hence, it has been necessary that the tests be carried out in short duration sessions.

## 6. Conclusions

This paper describes the computational tool for a communication system for quadriplegic patients, which includes a module that displays graphics components categories,

subcategories and phrases associated with the pictograms described. The interface also displays letters sorted based on their frequency of use, and suggested words based on selected letters. This computational tool is designed to allow a quick and intuitive way of communication to quadriplegic patients who lack their speech capabilities.

In order to validate the benefits and limitations of the proposed system, the graphical interface, integrated to an assistive communication system, has been subjected to tests by the patient for whom it was designed, and has gathered satisfactory results, as future work we are planning to use it in patients with different levels of disabilities, including several degrees of tetraplegia. For the above, efforts are under way to test the system in patients treated within the CRIT Sonora (Children's Rehabilitation Center in Sonora).

## 7. Acknowledgment

The first author wants to thank the financial support from CONACYT for the scholarship (grant 644447/571585). This study has been partially supported by the Tecnológico Nacional de México (TNM) (grant 5573.15-P).

## 8. References

- [1] INEGI. Censo de Población y Vivienda 2010: Tabulados del Cuestionario Ampliado. [Online]. Available: <http://www3.inegi.org.mx/sistemas/TabuladosBasicos/Default.aspx?c=27303> (Visitado: 19 de Enero del 2016).
- [2] S. Kirshblum, S. Burns, F. Biering-Sorensen, W. Donovan, D. Graves, A. Jha, M. Johansen, L. Jones, A. Krassioukov, M. Mulcahey, M. Schmidt-Read and W. Waring, 'International standards for neurological classification of spinal cord injury (Revised 2011)', The Journal of Spinal Cord Medicine, vol. 34, no. 6, pp. 535-546, 2011.
- [3] M. Brodwin, "Spinal Cord Injury," en Medical, Psychosocial and Vocational Aspects of Disability, 3ra ed. Elliot & Fitzpatric, Inc., 2009, pp. 289-304.
- [4] "OMS | La depresión" <http://www.who.int/mediacentre/factsheets/fs369/es/>, (Visitado: 26 de Enero de 2016).
- [5] J. Vázquez-Barquero Smith, 'La Clasificación Internacional del Funcionamiento, de la Discapacidad y de la Salud (CIF)', Organización Mundial de la Salud. 2001. [Online] Available: [http://conadis.gob.mx/doc/CIF\\_OMS.pdf](http://conadis.gob.mx/doc/CIF_OMS.pdf) (Visitado: 26 de Enero del 2016).
- [6] 'OMS | Discapacidades', 2015. [Online]. Available: <http://www.who.int/topics/disabilities/es/>. (Visitado: 26 de Enero del 2016).
- [7] INEGI. Clasificación de Tipo de Discapacidad. [Online]. Available: [http://www.inegi.org.mx/est/contenidos/proyectos/aspectosmetodologicos/clasificadoresycatalogos/doc/clasificacion\\_de\\_tipo\\_de\\_discapacidad.pdf](http://www.inegi.org.mx/est/contenidos/proyectos/aspectosmetodologicos/clasificadoresycatalogos/doc/clasificacion_de_tipo_de_discapacidad.pdf) (Visitado: 27 de Enero del 2016).

- [8] L. Simpson, J. Eng, J. Hsieh and D. Wolfe and the SCIRE Research Team, 'the Health and Life Priorities of Individuals with Spinal Cord Injury: A Systematic Review', *Journal of Neurotrauma*, vol. 29, no. 8, pp. 1548-1555, 2012.
- [9] R. Areny, *Sensores y acondicionadores de señal*, 4th ed. Barcelona: Marcombo, 2005, pp. 1-8.
- [10] M. Beigl, A. Krohn, T. Zimmer and C. Decker, 'Typical Sensors needed in Ubiquitous and Pervasive Computing', *Proceedings of INSS*, pp. 22-23, 2004.
- [11] 'Products – The Eye Tribe', 2015. [Online]. Available: <https://www.theeyetribe.com/products/>. (Visitado: 01 de Febrero del 2016).
- [12] N. Garay, J. Abascal and L. Gardeazabal, "Mediación emocional aplicada en sistemas de comunicación aumentativa y alternativa", *INTELIGENCIA ARTIFICIAL*, vol. 6, no. 16, 2002.
- [13] GazeSpeaker, 'GazeSpeaker - Speaking with the eyes', 2015. [Online]. Available: <http://www.gazespeaker.org/>. [Visitado: 16 de Marzo de 2016].
- [14] O. Grijalva, M. Serna-Encinas, C. Rose, O. Rodriguez-Elias, J. Islas, 'Design of a Graphical Interface for a Communication System for Persons with Tetraplegia Using an Eye-Tracking Sensor' 2016 4th International Conference in Software Engineering Research and Innovation (CONISOFT), Puebla, 2016, pp. 33-38.
- [15] "Portal Aragón de la Comunicación Aumentativa y Alternativa" <http://www.arasa-ac.org/index.php/>, (Visitado: 28 de Enero de 2016)
- [16] "Real Academia Española - CREA" <http://corpus.rae.es/lfrecuencias.html/>, (Visitado: 28 de Enero de 2016).

# Chapter # 5

## Driving Security Aware Android Application Development Based on Malware Analysis Data Visualization

A. Rodríguez-Mota\*, P.J. Escamilla-Ambrosiot, E. Aguirre-Anayat,  
R. Acosta-Bermejot and L.A. Villa-Vargast

\*Instituto Politécnico Nacional

Escuela Superior de Ingeniería Mecánica y Eléctrica, Zacatenco, México City, México 07738

Email: armesimez@gmail.com

† Instituto Politécnico Nacional, Centro de Investigación en Computación, México City, México

Email: pescamilla@cic.ipn.mx

## 1. Introduction

Android is an operating system for mobile phones. However, it can be better described as a middleware running on top of embedded Linux. The underlying Linux internals have been customized to provide strong isolation and contain exploitation. Each application is written in Java and runs as a process with a unique UNIX user identity. This design choice minimizes the effects of buffer overflows. In this structure, all inter-application communication passes through Android's middleware, with a few exceptions [1].

The Android middleware defines four types of interprocess communication (IPC). The types of IPC directly correspond to the four types of components that make up applications. Figure 1 describes the main Android components. Generally speaking, IPC takes the form of an Intent message. Intents are either addressed directly to a component using the application's unique namespace, or more commonly, to an action string. Developers specify Intent filters based on action strings for components to automatically start on corresponding events [1].

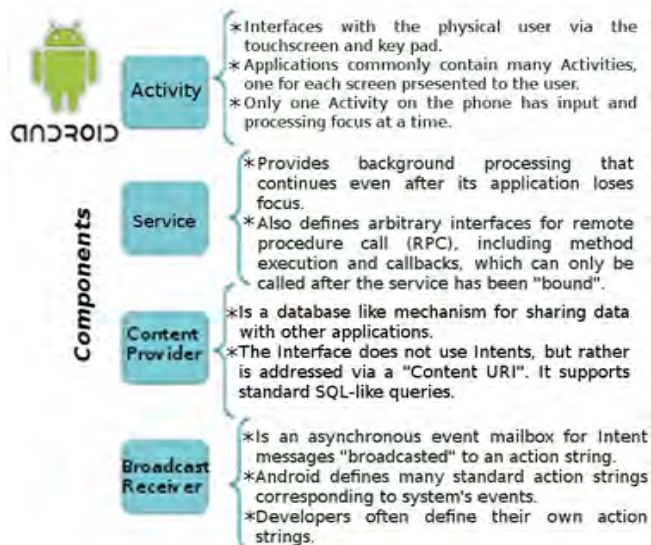


Figure 1. Main Android components, adapted from [1] as presented in [2].

The Android build process provides project and module build settings so that Android modules are compiled and packaged into .apk files, the containers of application binaries, based on the developer build settings. The apk file of each application contains all of the information necessary to run an application on a device or emulator [3]. Figure 2 shows the main contents of an apk file.



Figure 2. Main contents of an apk file.

## 2. Android security

Android protects applications and data through a combination of two enforcement mechanisms, one at the system level and the other at the IPC level, Figure 3 provides a general representation of such mechanisms. IPC mediation defines the core security framework but it builds on the guarantees provided by the underlying Linux system [4].

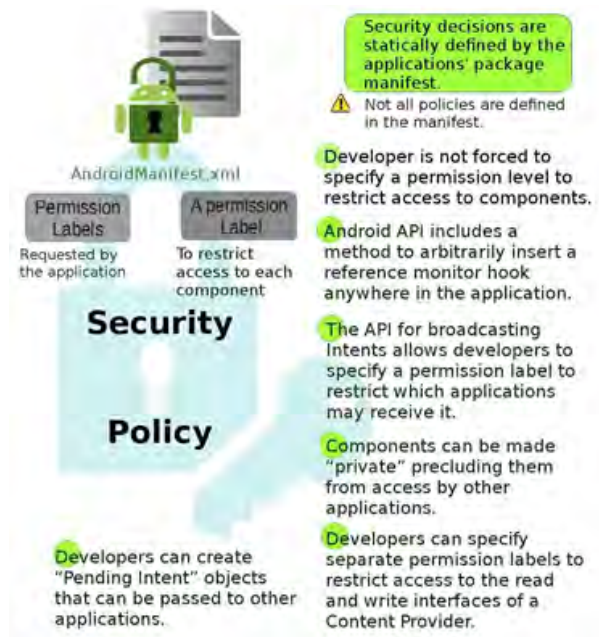


Figure 3. General representation of Android's security enforcement mechanisms, adapted from [1].

In the general case, each application runs as a unique user identity, which lets Android limit the potential damage of programming flaws. IPC is not limited by user and process boundaries. In fact, all IPC occurs via an I/O control command on a special device node, /dev/binder. Because the file must be world readable and writable for proper operation, the Linux system has no way of mediating IPC. Although user separation is straightforward and easy understood, controlling IPC is much more subtle and warrants careful consideration [4].

As the central point of security enforcement, the Android middleware mediates all IPC establishment by reasoning about labels assigned to applications and components. A reference monitor provides mandatory access control (MAC) enforcement of how appli-

cations access components. In its simplest form, access to each component is restricted by assigning to it an access permission label. Developers assign applications collections of permission labels. When a component initiates IPC, the reference monitor looks at the permission labels assigned to its containing application and allows IPC establishment to proceed if the target component's access permission label is in that collection. The developer assigns permission labels via the XML manifest (`AndroidManifest.xml`) file that accompanies every application package, see Figure 2. Because Android's policy enforcement is mandatory, all permission labels are set at install time and cannot change until the application is reinstalled [4].

The Android community has made huge progress in improving security. Features such as full-disk encryption, restricted profiles, improved authentication and SafetyNet all give Information Technology (IT) shops better Android device management and security capabilities. But IT manager would be well-served to pay careful attention to the potential pitfalls that remain. In this context, five main Android device security challenges to focus on in the coming years are fragmentation, malware, management tool selection, user behavior, and compartmentalization [5].

## 2.1 Fragmentation

Perhaps the single biggest criticism that Android has received has to do with the diversity of its ecosystem. As an open source OS, Android has a wide range of modified versions implemented on a significant number of devices [5]. A solution to the Android fragmentation problem is to limit the number of devices and operating system versions allowed, whether they are corporate-issued or Bring Your Own Device (BYOD). These approaches facilitate IT control over Android device management and security in production environments [5].

Fragmentation is a serious problem with Android devices. Each manufacturer puts its own spin on a device, providing features and configuration options different from other Android devices. Even if all devices are running the latest Android version (currently Android 6.0), administrators must still contend with vendor-specific hardware features and software tweaks [6].

## 2.2 Malware

One of the most effective solutions to the problem of malware is the use of mobile application management, which can prevent an infected app from contaminating an entire device. Acceptable use and security policies need to be in place before deploying

any management products, however, these policies must align with overall organizational objectives. Antimalware apps are available, but their effectiveness is controversial, and IT should carefully set them before deployment commences [5].

## 2.3 Management tool selection

Enterprise mobility management (EMM) suites can help improve Android device security with their content, application and identity management features. The key elements to look for are cross-platform support, especially across multiple Android releases, and integration with other operational management systems. Those capabilities are becoming more important to avoid overlapping or conflicting features, as well as to maximize IT productivity [5].

## 2.4 User behavior

Encouraging users to comply with simple, straightforward policies can solve many Android device security problems. But if even one user fails in this regard, it can lead to quite the opposite. Policies should require the use of device passcodes, appropriate backup and storage and adherence to best practices for avoiding social engineering attacks. The biggest problem area regarding user behavior has to do with apps. Even though Android apps provide permissions notifications, mobile content and application management are clearly more critical than ever to control the flow of data among apps; and do not hesitate to blacklist third-party apps that raise security concerns [5].

## 2.5 Compartmentalization

In the case of BYOD situations, users expect to do whatever they want with what are, after all, their own devices. EMM is the backbone of good organizational security practice for now, but dual persona and mobile virtualization which separate a single device into separate work and personal environments will become more common. The good news is that Android device management supports these technologies, where iOS does not [5].

The open source nature of Android should provide some comfort that the architecture and security mechanisms of a given implementation are appropriate, effective and uncompromised. Many, however, remain skeptical of open source in general and of Android in particular, thanks to past problems with app security. Vigilance remains a core requirement in all IT departments [5].



### 3. Malware Analysis

Malware analysis is a process in which malware is taken apart for studying its code structure, operation and functionality. It is conducted with specific objectives which include: to understand the vulnerability that was exploited, to study the severity of the attack and counteracting measures, to penetrate into the compromised data in order to investigate its origin and to obtain information about other compromised machines [7].

In general, detection techniques for Android malware use statically extracted data from the manifest file or from Android API function calls, as well as dynamically obtained information from network traffic and system calls tracing [8]. Most of current systems used to detect malicious code are largely based on syntactic signatures and employ static analysis techniques.

Privacy-violation weaknesses occurring on mobile devices can lead to the disclosure of location, sensitive images, and data entered from the keyboard or displayed on the screen and other personal information. While smartphones can be used for viewing, manipulating, and storing local data, these devices also allow users to interact with a world of interconnected resources from the convenience of their hands. Through communication protocols, both sensitive and benign data is shared between remote services in different devices [9]. In the context of Android, privacy violation weaknesses can be related to a set of security risks, Figure 4 presents 10 of the biggest Android security risks.

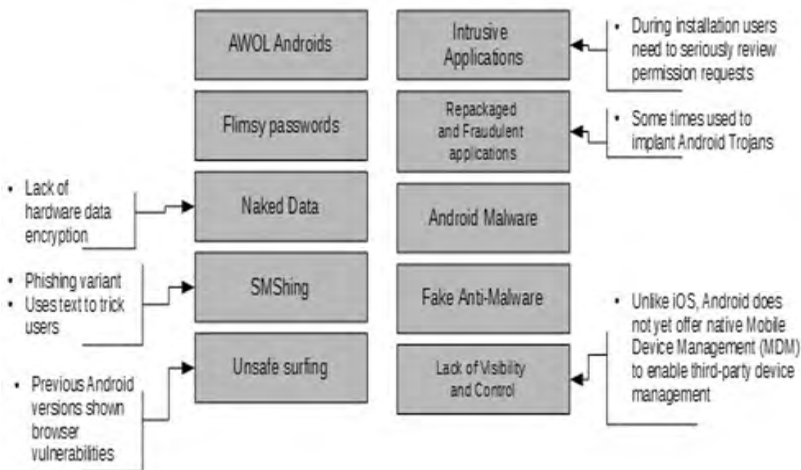


Figure 4. Android Security Risks [10], as presented in [11].

### 3.1 Analysis Techniques

Malware analysis and detection schemes can be deployed on both mobile devices and servers. On the one hand, client side security solutions include anti-virus or antimalware applications installed on mobile devices to protect against known applications installed on mobile devices based on knowing signatures of malicious applications. However, installing an application to provide real time protection on a mobile device often decreases its performance and battery life [12]. On the other hand, remote or cloud-based solutions are designed to offload a significant part of their operation to the cloud, where their computationally intensive algorithms and analyzes are running. Although this type of mechanisms saves system resources, on their own, they cannot offer real-time protection, and they can leave devices vulnerable when connectivity with the server is poor [13].

### 3.2 The Hybrid Approach

The implementation of hybrid solutions for malware analysis and detection is not a new approach in the PC antimalware arena. However, only a few attempts to explore such approaches have been reported in the malware detection for mobile device literature [14]. In this sense, it can be considered that there is a big research area to explore in this field. Consequently, in this chapter, it is discussed an approach that has been referred to as “2-hybrid”. The term 2-hybrid is used in order to reflect the intentions to provide local (host) and remote (server/cloud) implementation and static dynamic analysis capabilities. This scheme aims to provide a balanced and efficient analysis-and-detection framework in terms of time, resources consumption and performance. A conceptual representation of the intended system is presented in Figure 5.

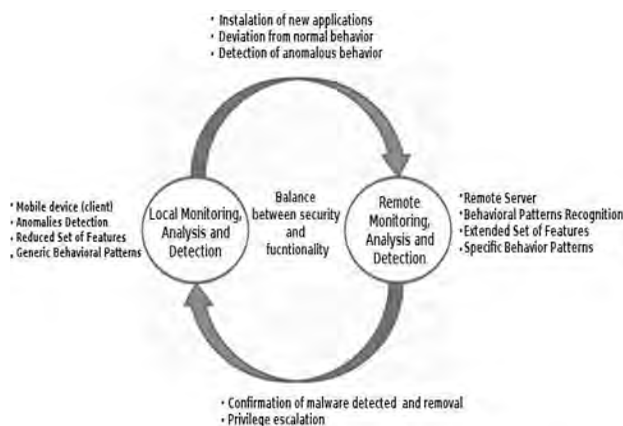


Figure 5. The 2-hybrid system general conceptual representation [14].

## 4. Android development security checks

As the number of threats has increased over the time Android OS has matured, Google and the Open Handset Alliance (OHA) have led many efforts towards reducing the impact of malicious attacks targeted to this OS. Interestingly, developers following bad software development practices continue to be an important risk factor, mainly due to the lack of knowledge or misuse of Android security features.

In the particular case of the Androids permission model, this is a complex situation as developers are able to add permissions and access features of the mobile device if needed; these features and permissions most often add value to the application. However, developers, using the same permissions that create value for the applications, may also create malicious applications that can harm users or their devices. Unfortunately, the only solution for checking if an application is malicious or not is to carry out a sanity check. It means the user should read carefully what the application does and check the permission list to identify if the application really does need the permissions to work properly.

Many users do not know much about the underlying technology being used and what permissions are needed for certain features [15]. Tables I, II, and III, present some Android permissions that can harm the functionality of other applications, operating systems or hardware sensors.

### 4.1 Promoting Security Best Practices

Mastering the use of permissions is not a simple task, even for developers, as it requires a profound Android specification knowledge. Various efforts have been made by different entities, aiming to define and communicate the main security considerations that would help to produce more secure applications, as well as to improve user security practices. Some of these initiatives are briefly presented in the following paragraphs.

- *Android Security Best Practices*: The Android Security Best Practices document [16] provides a comprehensive view of the diverse security aspects to be considered during application development. A synthesis of such guidelines is shown in Figure 6. In some cases, these practices require the review of some static parameters either at the level of the `AndroidManifest.xml` file or the Java code. Some of these parameters have been represented as light gray colored boxes in Figure 6.

- OWASP Mobile Application Security Guide: The Open Web Application Security Project (OWASP) has proposed a set of security guidelines which are contained in the Mobile Application Security Guide document [17]. In this case, the check list contains general aspects applicable to most mobile devices, as well as a set of considerations specific to the Android OS powered devices. Figure 7 presents a synthesized representation of the static check points of this Security Guide, specific to Android. In this figure it has been included the “Store Sensitive Information outside App Sandbox”, highlighted with a light gray background, which the Application Security Guide mark as a feature that is obtained dynamically. This aspect is included in the figure, although marked as dynamic in the Guide, as it is possible to identify some permissions, as well as few static parameters, which grant the application access to external storage devices. After comparing Figures 6 and 7, it can be observed that the OWASP Guide document is more general than the Android Security Best Practices.

Table 1. A list of some android permissions that can harm the functionality of other applications, operating systems or hardware sensors, adapted from [15].

Permission	Related Risks
Change Network State	Can change whether or not the device is connected to a network.
Keyguard	Can keep the Android device unlocked and unprotected, causing unwanted calls.
Modify Audio Settings	May affect the usability of the device or impact other applications.
Set Time Zone	May affect the usability of the device or impact other applications.
Write External Storage	May result in a number of effects being realized including harming the actual memory of the device. Many writes and deletes may break memory segments. An application may fill the devices memory storage such that a victim would be unable to add more data or install required applications.
Write Contacts	Similar effects as WRITE EXTERNAL STORAGE. By adding contacts malware could trick the user into calling unwanted numbers, change phone numbers of certain contacts or by adding contacts malware could fill the space for contacts on a SIM card.
Kill Background Processes	It is not necessarily malicious and would not do much harm, but it could impact the usability of the device, for example by killing processes without user's consent.

- *CIS Security Concerns*: The security concerns addressed by the Center for Internet Security (CIS) [18] includes 41 main settings that, according to this document, users can and should configure on their Android devices. These settings can be divided into seven distinct categories, see Figure 8. Under comparison, this document has a more general perspective when compared with the others two presented above, but it provides a good reference point..

**Table 2. A list of some android permissions that can harm the functionality of other applications, operating systems or hardware sensors (continuation), adapted from [15].**

Permission	Related Risks
Set Wallpaper	It is not necessarily malicious and would not do much harm, but it could impact the usability of the device.
Vibrate	It is not necessarily malicious and would not do much harm, but it could impact the usability of the device.
Call Phone	Used to call a phone number. Is one of the most used permissions in revenue generating malicious applications? Malware will just randomly or on a certain event, call a premium number.
Process Outgoing Calls	Is able to intercept an outgoing call and change it. This could be used maliciously to redirect all calls to some premium number.
Send SMS	It can be used by attackers in their applications and send messages to premium numbers.
Send MMS	It can be used by attackers in their applications and send messages to premium numbers.
Write SMS	May be used for writing SMS, however, the SMS may not be sent without the user's confirmation. It is safer if an application has this permission, but the victim could still be tricked to execute the sending of the message.
Internet	It is no dangerous by itself, it allows access to the Internet for applications, but can be used for transfer of personal data, documents or files of the victim.
Get Accounts	Allows access to the list of accounts in the account service. This is a lower risk permission than the MANAGE ACCOUNTS permission. An application can just get the basic properties of an account, such as the user name.

Table 3. A list of some android permissions that can harm the functionality of other applications, operating systems or hardware sensors (continuation), adapted from [15].

Permission	Related Risks
Manage Accounts	An application using this permission can get more than basic properties of an account and is able to do all the needed actions with existing accounts and to add new accounts. As a Google security measure an application may only delete/modify an account it created itself, an application may, of course, create any new account and manage that.
Use Credential	Is used to log into an account. In most cases, “credentials just means the corresponding authenticator creates a fitting token and hands that over.
Manage Documents	Allows an application to manage access to documents. This permission can give access to documents that can be then read.
Read External Storage	Used for reading files.
Read Sms	Used for reading SMS. This permission allows access to SMS log and works in conjunction with the RECEIVE SMS permission.
Receive Sms	Enables the application to be notified and read the incoming SMS message.
Read Call Log	Allows an application to read the call log. Work in conjunction with the PROCESS OUTGOING CALLS permission.
Read Contacts	Allows an application to read the list of contacts from the mobile device.
Read Profile	Allows an application to read the user’s personal profile data.

After reviewing this document, it is possible to identify multiple coincidences between these initiatives, among them the Permissions requests aspect.

## 5. Garmdroid

GARMROID, is a 2-hybrid Android malware analysis and detection system under development [14]. Its name is the result of the fusion of the words Garm and Android, where Garm is described in the Norse mythology as a watchdog that guards Hels gate [19]. Broadly, the proposed system is categorized as 2-hybrid to reflect the fact that it integrates static and dynamic malware analysis techniques and local and remote analysis execution. GARMROID general concept is presented in Figure 9. Additionally, Figure 10

presents GARMDROID current architecture. A detailed description of the system design and implementation is out of the scope of this chapter.

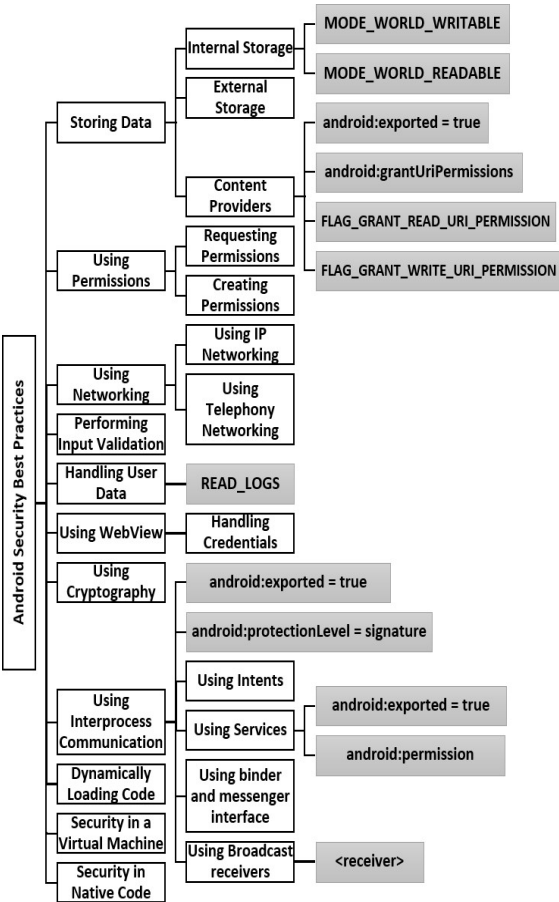


Figure 6. Android security best practices, adapted from [16].

GARMDROID functionality is based on the extraction of static and dynamic mobile malware features. A taxonomy of mobile malware features is presented in Figure 11.

Since the scope of the proposed system is broad, as it can be inferred from the set of mobile malware features in Figure 11, the current description is focused on the utility of extracting and analyzing the Permissions and Hardware components static features of applications, aiming to help software developers to identify security risks and bad practices for Android application development. Figures 12, 13, and 14 present the GARMDROID components related to the extraction and visualization of permissions and hardware

features. In this respect, it is important to consider that several research articles have shown that users respond better to visual information, in terms of Android permissions and other contexts [21, 22, 23, and 24].

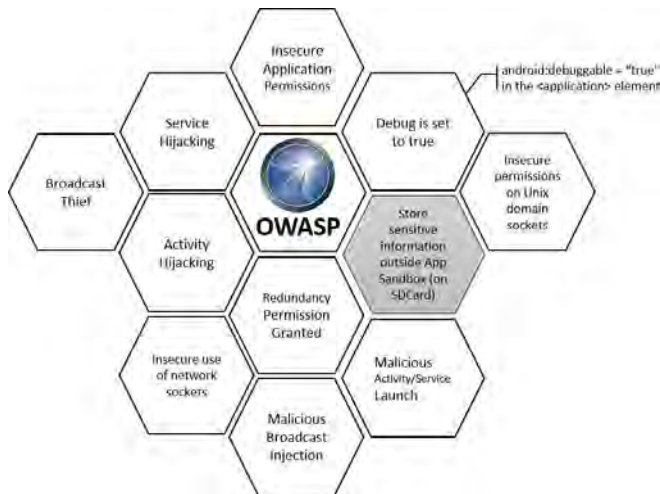


Figure 7. Selected OWASP Static (Client side) checks, adapted from [17].

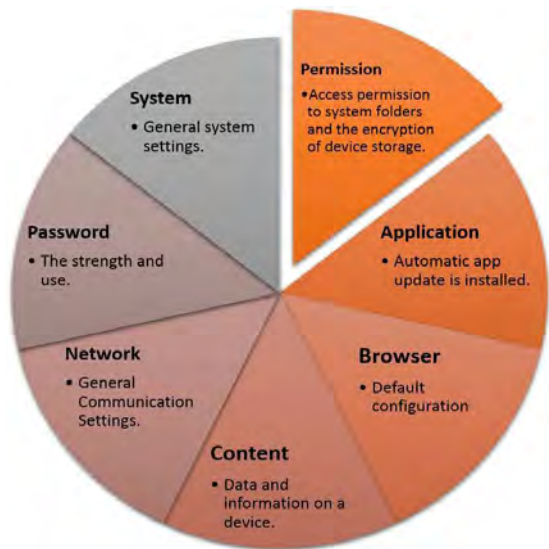


Figure 8. The seven categories of settings that Android users should configure, according to the Center for Internet Security (CIS) Security Concerns description, adapted from [18].



## 6. Results

A set of results are presented in this section with a twofold purpose: to demonstrate GARMDROID operation and to provide evidence of the utility of the obtained results for identifying security threats related to malicious or bad design practices. A test version of GARMDROID is available at [www.garmdroid.org](http://www.garmdroid.org).

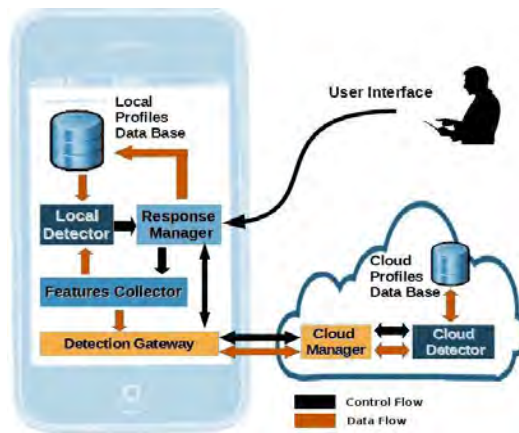


Figure 9. General representation of the proposed 2-hybrid malware analysis and detection framework [14].

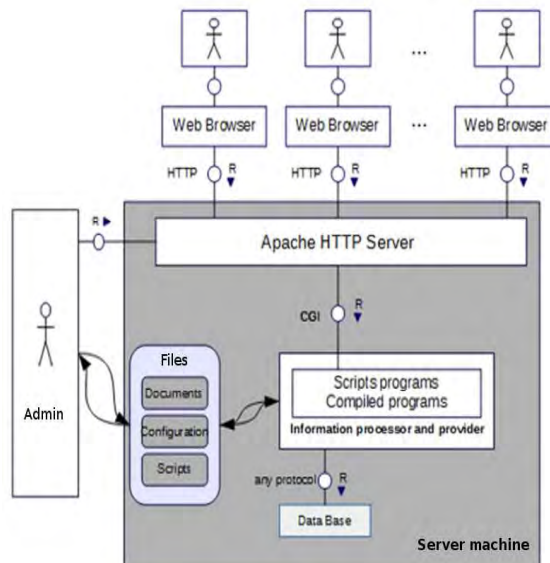


Figure 10. GARMDROID system architecture.

Brief descriptions of the three selected cases are presented in Table IV. The descriptions refer to Android applications identified as benign when analyzed through VirusTotal [25] free service. It is important to observe that they were downloaded from third-party stores on the Web.

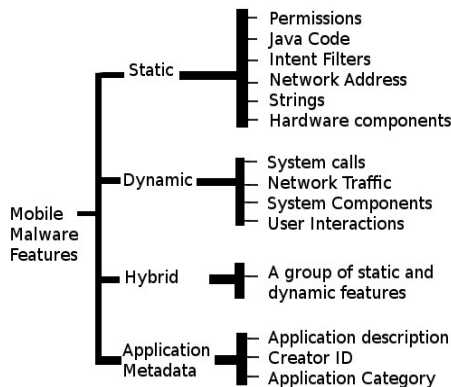


Figure 11. Taxonomy of Mobile Malware Features [14], adapted from [20].

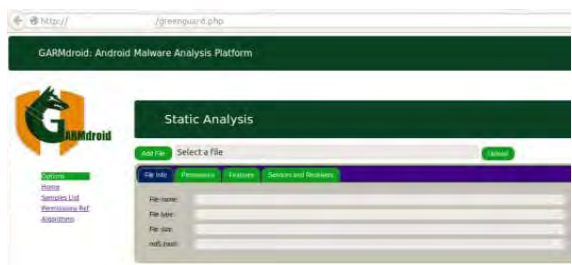


Figure 12. GARMdroid welcome page.

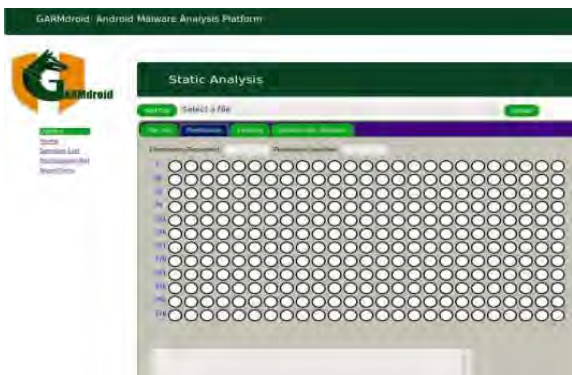


Figure 13. Permissions representation using an array of dots for simple visualization, and a text box.

A. **Hardware-Test app**

In this case, see Figure 15, the app requests access to a big set of hardware-features: Accelerometer, Audio, Barometer, Bluetooth, Camera, Compass, Gyroscope, Light, Location, Microphone, NFC, Proximity, Scree, Telephony,

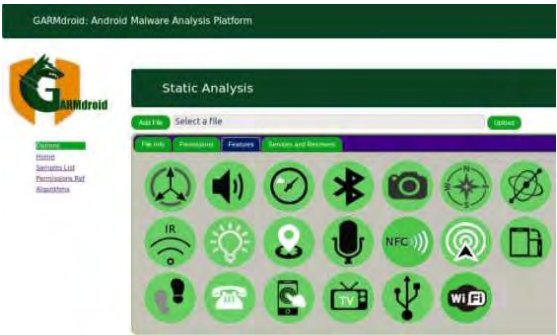


Figure 14. Visualization of hardware features.

Table 4. Application analysis sample cases.

Application Type	Risk observed
Hardware-Test	Granting a big set of permissions and access to many hardware features represents a high security risk.
IR-remote control	Excessive hardware-features requests represent a sign of malicious intents or bad design practices.
Lighting	Inconsistency between the advertised functionality of an application and the hardware features requests must raise security concerns.



Figure 15. Hardware features requested by a hardware-test sample app, marked with a red background.

## Touchscreen, USB and Wi-Fi

In general, it could be observed from these results that there are not inconsistencies between the requested hardware-features and the supposed functionality. However, developers and system integrators must be aware of the high risk involved when installing applications requesting a big set of permissions.

### B. IR remote control app

The second selected case presents a pair of apps advertised as IR remote controls in the application store, see Figure 16.



Figure 16. Hardware features requests from two different IR remote control apps.  
Requested features are marked with a red background.

In this scenario, it is observed that the number and type of requests of hardware-features done by the first app, (a) in Figure 16, includes not only the IR feature but Bluetooth, Wi-Fi, Screen, and Touchscreen too. It can be argued that remote control functions can be provided through the use of Bluetooth and Wi-Fi features too, but either from a security or software development point of view there exist inconsistencies between the functionalities provided by the app and those required (or advertised). This should raise questions in turn of whether the design specification was not properly followed, bad development practices have drawn the software development process or there is a malicious intention.

### C. Lighting app

In this final case, the requested hardware-features by a Lighting app are presented, see Figure 17, where once again inconsistencies are observed between the presuming functionality and the hardware requests. Besides the Camera, it is observed that this

application requests location and WiFi features. Thus, there exists either a bad description of the app in the store, bad design practices or malicious intentions in this piece of software.

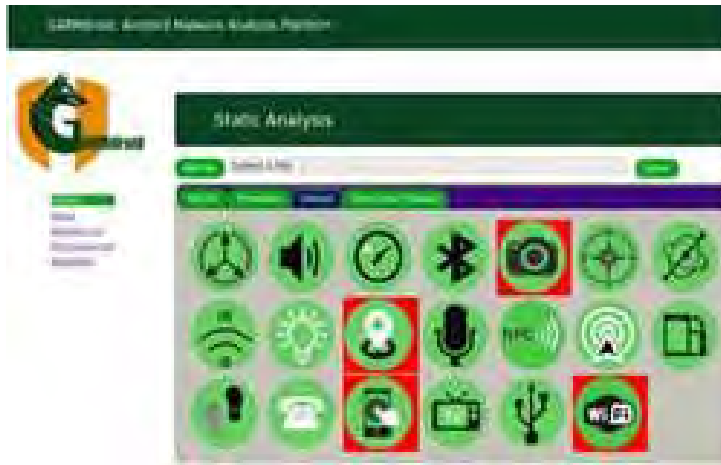


Figure 17. Hardware features requested by a lighting application.

## 7. Conclusions

On the one hand, the high demand for Android applications combined with the fast expansion of Android OS in the mobile market has contributed not only to attract the attention of malicious developers and criminals but to force genuine developers to adopt this technology at a fast pace. In this sense, security has been neglected in order to prioritize delivery and achievement of short time releases.

On the other hand, the current increase in demand for solutions to improve security in mobile applications has produced a continuous development of tools and techniques aimed to prevent or at least to reduce security threats in this technological segment. However, security analysis and detection practices are hardly adopted by developers and managers. One could argue that complexity or the need of specialized knowledge may play an important role in the slow adoption of security practices into the mobile software development processes. But the daily practice has shown that there is space to perform some level of security tests during development and systems integration by providing easy to use tools or higher level representations of security information more comprehensible for non-specialized users. In this context, the development of systems and tools which provide a higher representation of security analysis data seems a good option to permeate security practices into the software development process and con-

sequently reduce bad practices and possible vulnerabilities left unintentionally. Thus, we believe that the architecture and functionalities proposed in this work contribute to bring security data analysis and practices to a higher number of developers and other subjects involved in the Android mobile software development only requiring an Internet connection.

Additionally, it is considered that the results presented, represent a good probe of the utility of malware analysis and detection data as an important asset for software developers, helping to reduce bad practices and design errors. Moreover, these results give an indication that there is need for further development of high level visual representations of security data with affordance and easy to use qualities.

Finally, as GARMDROID is still a work in progress, it is considered that further integration of other malware analysis and detection techniques will provide new insights about data visualization and malware detection that will help to refine current approaches and to develop new ones.

## 8. Acknowledgment

This work was supported by the Mexican National Council of Science and Technology (CONACYT) under Grant 216747 and in part by IPN under Project SIP-20161697.

## 9. References

- [1] W. Enck, M. Ongtang, and P. McDaniel, On Lightweight Mobile Phone Application Certification, CCS'09 Proceedings of the 16th ACM conference on computer and Communications Security, ACM, New York, NY, USA, 2009, pp. 235-245.
- [2] A. Rodríguez-Mota, P. J. Escamilla-Ambrosio, E. AguirreAnaya, R. Acosta-Bermejo and L. A. Villa-Vargas, Improving Android Mobile Application Development by Dissecting Malware Analysis Data, 4th International Conference in Software Engineering Research and Innovation (CONISOFT), 2016, pp. 81 - 86.
- [3] Android Developers, Building and Running Overview, <http://developer.android.com/intl/es/tools/building/index.html>, 29 1 2016.
- [4] W. Enck, M. Ongtang and P. McDaniel, Understanding Android Security, Security & Privacy, IEEE, vol.7, no.1, Jan.-Feb. 2009, pp. 50-57.
- [5] C. Mathias, Top five Android device management security challenges, TechTarget, <http://searchmobilecomputing.techtarget.com/tip/Top-Five-Android-device-management-security-challenges>, 2 2 2016.

- [6] R. Sheldon, Android OS fragmentation curbs enterprise adoption, TechTarget, <http://searchmobilecomputing.techtarget.com/tip/Android-OS-fragmentation-curbs-enterprise-adoption>, 22 2016.
- [7] K. Kendal and C. McMillan, Practical Malware Analysis. Black Hat Conference, USA, 2007.
- [8] V. M. Afonso, M. Favero de Amorim, A. R. A. Gregio, G. Barroso Junquera and P. Lício de Geus, Identifying Android malware using dynamically obtained features, Journal of Computer Virology and Hacking Techniques, Springer, vol. 11, no.1, February. 2015, pp. 9-17.
- [9] D. Childs, A. Gilliland, B. Gorenc, H. Goudey, A. Gunn, A. Hoole, J. Lancaster, S. Muthurajan, J. Wook Oh, Y. Tsipenyuk O'Neil, J. Park, , O. Petrovsky, J. Sechman, N. Shah, T. Sotack and V. Svajcer, The HPE Cyber Risk Report 2015. HP, 2015.
- [10] L. Phifer, Top 10 Android Security Risks, <http://www.esecurityplanet.com/views/article.php/3928646/Top-10-Android-Security-Risks.htm>, 14 05 2015
- [11] A. Rodríguez-Mota, P.J. Escamilla-Ambrosio, J. Happa and E. Aguirre-Anaya, GARM-DROID: IoT Potential Security Threats Analysis through the Inference of Android Applications Hardware Features Requirements, AFI 360 Conference Track on Future Internet and Internet of Things Applications, 2016
- [12] N. Penning, M. Hoffman, J. Nikolai, and Y. Wang, Mobile malware security challenges and cloud-based detection, Collaboration Technologies and Systems (CTS, 2014 International Conference, 2014, pp. 181-188.
- [13] D. Damopoulos, G. Kambourakis and G. Portokalidis, The Best of Both Worlds: A Framework for the Synergistic Operation of Host and Cloud Anomaly-based IDS for Smartphones, Proceedings of the Seventh European Workshop on System Security, ACM, New York, NY, USA, 2014, pp. 6:1-6:6.
- [14] A. Rodríguez-Mota, P.J. Escamilla-Ambrosio, S. MoralesOrtega, M. Salinas-Rosales and E. Aguirre-Anaya, Towards a 2-hybrid Android Malware Detection Test Framework, 2016 International Conference on Electronics, Communications and Computers (CONIELECOMP), Cholula, 2016, pp. 54-61.
- [15] N. Milosevic, Android Security: Malicious use of Android permissions, Digital Forensics Magazine. Issue 18, February, 2014, pp. 28-31.
- [16] Android developer, Security Tips, <http://developer.android.com/training/articles/security-tips.htm#Dalvik>, 14 06 2016.
- [17] OWASP, OWASP Mobile Application Security Guide, <https://drive.google.com/file/d/0BxOPagp1jPHWYmg3Y3BfLVhMcmc/view?pref=2&pli=1>, 5 05 2016.
- [18] D. Vecchiato, M. Vieira and E. Martins, The Perils of Android Security Configuration, Computer, IEEE, 2016, Vol. 49, No. 6, pp. 15-21.
- [19] Wikipedia, Garmr. Wikipedia, <https://en.wikipedia.org/wiki/Garmr>, 15 12 2015.

- [20] A. Feizollah, N.B. Anuar, R. Salleh, and A. W. A. Wahab. A review on feature selection in mobile malware detection. *Digital Investigation*, Elsevier, 2015, pp. 22-37.
- [21] J.R.C. Nurse, I. Agraftotis, M. Goldsmith, S. Creese and K. Lamberts, Two Sides of the Coin: Measuring and Communicating the Trustworthiness of Online Information, *Journal of Trust Management*, Vol. 1, No. 1, pp. 1-20, 2014.
- [22] L. Krauss, I. Wechsung and S. Mller, Using Statical Information to Communicate Android Permission Risks to Users, 4th International Workshop on Socio-Technical Aspects in Security and Trust (STAST'14), Vienna, Austria, 2014, pp. 48-55.
- [23] C. Eze, J.R.C. Nurse and J. Happa, Using Visualizations to Enhance Users' Understanding of App Activities on Android Devices, 2016, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, Innovative Information Science and Technology Research Group, Vol. 7, pp. 39-57.
- [24] M. Hettig, E. Kiss, J. Kassel, S. Weber, M. Harbach, and M. Smith, Visualizing Risk by Example: Demonstrating Threats Arising from Android Apps, *Symposium on usable Privacy and Security (SOUPS'13)*, Newcastel, UK, 2013, pp. 1-2.
- [25] VirusTotal. <https://www.virustotal.com/es-mx/>.



# Chapter # 6

## A Methodology to Assist Novice Engineers to Produce Quality Research and Development Projects

Josefina Guerrero-García<sup>1</sup>, Juan González-Calleros<sup>1</sup>, Jaime Muñoz-Arteaga<sup>2</sup>,  
Arturo Morales<sup>1</sup> e Ivonne Monarca<sup>1</sup>

1 Facultad de Ciencias de la Computación, Benemérita Universidad Autónoma de Puebla

2 Facultad de Ciencias de la Computación, Universidad Autónoma de Aguascalientes  
jguerrero,juangonzalez{@cs.buap.mx}, jmunozar@correo.uaa.mx

### 1. Introduction

Producing quality products is always desired when we start a project. A quality product is assumed to be the one that has the expected functionality but also satisfies users expectations. With regard to the functionality, the development must focus not just on the running product but also in the documentation describing the product. For decades this has been a problem for undergraduate students or naïve engineers [1, 5]. Some of the common problems reported in the literature are related to: bad use of name conventions or semantics [1, 5, 7, 14, 15, and 16], misconceptions of abstraction levels [1], cardinality [1], tools and visual syntax [2,8], evolutionary design missing [1], cognitive load to differentiate concepts [5,9], limited information processing capabilities [6]. If people are not good at it, why we keep promoting their use? So authors argue that they communicate information more effectively than text [5], an assumption that we keep considering while teaching but with poor results while evaluating students.

As reported in [5, 7, 14, 15, and 16], the lack of semantics is one of the common problems. Current mapping notations are not clear to what real world they have a correspondence. Existing metaphors are not easy to understand, thus creating and reading diagrams is not easy for novice learners due to the cognitive load. Attention must be paid to the usability of symbols used in the notations [6, 16] trying to be more cognitive effective [7].

In a previous work [30] we presented our findings of three years of research with senior year students while working on a one-year project. With the following research questions: is it possible to improve the quality of the visual notations for analyzing and design software, by empowering students with decision making about what to use and how to use it? As promoted in an agile paradigm, empowering teams is a recommended practice for a successful project. The only condition is to have an agreement with the whole team. The goal is to help them to create quality models and contributing to have a better understanding of the notations among our community.

For this purpose, we conducted a set of experiments, the first year we just simply guided students towards the correct use of existing notations and adopting a waterfall methodology for the development of the project. Notations used during our research was: use cases, state-charts, class diagrams, Sequence diagrams, activity diagrams or other process models such as BPMN or Petri Nets, Task Models, and UI prototyping notations. During the second year, we change the development strategy and start on relying on an agile methodology without flexibility to modify the notations. While the improvement in the quality of models was perceived yet most of the students were not able to fully understand them. Finally, during the third year, we kept the agile methodology SCRUM, and let the teams modify or propose their own notation with really interesting diagrams that at the end were easy to understand not just by the whole team presenting the project but by the rest of their colleagues. Those diagrams even promote deep discussion about software and information architecture.

In this chapter, we extend the previous work [30] to explain in more details the proposed methodology with the importance of empowering end-users in the process and let them producing. The lesson learned on how they modify their attitude to create quality documentation and how happy they are nowadays by applying it at work.

## 2. State of the Art

Problems with current notations for the design of software has been reported for business process modeling [6, 7], UML [1, 4, 5, 11, 14, 17, and 21]. Most of these works agree on the need to reconsider current graphical notations for the design of software or to extend existing ones [19, 24, 25, and 26].

Adding visual aids to graphical notations for the design and modeling of software has been reported in the literature. Some works propose the use of colors to reduce the cognitive load while reading complex diagrams, for instance, UML [4] or graph transformations [19]. The use of colored aid proves to be beneficial, however, we have a limited

capacity not just to perceive colors (some people is color-blind), but also we can not differentiate a high amount of colors, research has proved that we can deal just with 4 chunks of information at a time [22]. So the solution is limited to problem where with limited variations.

Modifying the graphical notations is something that has been widely used in the literature. Trying to address limitations of current notations to represent agent-based systems [23], hypermedia [24], security [25], workflow [26]. While adding variants to existing notations is useful, some people argue that by doing that, you might kill the original purpose of the notations and overload the cognitive load to understand it [27]. Even worse you are on the risk that the variant proposal to a notation might not be adopted by a community thus it might demand a lot of effort to keep updated, if we develop a software or create a plug-in of an existing software, of new graphical notation. Using the right notation for the right model [27], also represents a challenge when the notation does not exist at all, for instance, how to design an auditory user interface? Or when the notation lacks expressiveness, for instance, in UML labeling method and class names with Italics to denote abstract Classes is not easy identified, sometimes, not even supported in existing solutions.

Some communities work to address the problem of lack of agreement or expressiveness in the notations. This is case of the Model-Driven Engineering community [28] that reach some agreements on what concepts the notations must include but so far, no agreement on the graphical notation exist. A similar effort was conducted for agent-based software [15] but contrary to previous example, they now are at the stage of validating the cognitive load of the i\* graphical notation but no further effort has been reported.

### **3. The Research: On the Limitations of Modeling in Software Engineering**

Software Engineering is an activity that uses diagrams to interchange knowledge among the different stakeholders that produce software. However, contrary to what you expect, diagrams are normally relegated by the software development community, based on our interviews, for different reasons: lack of time to create the diagrams, lack of knowledge to create or read them, lack of interest as their utility seems to be not relevant [21]. This perspective was collected from interviews with different stakeholders of the software development industry and after some years of dealing with real-life projects and handling our own company. But contrary to what people might think the true is that there is no way to keep alive a project without the right documentation. Let us introduce the problem with the following scenario, from an interview: our subject told us: “we are mi-

grating software made in C to Java but there is no documentation at all about how the system was built even worse the code is vaguely commented, after two months of work the company considered that doing from scratch the software would be the best solution, but surprisingly, one month after we start, the project was cancelled". You might think that perhaps we are referring to an inexperienced company but this is not the case. We are talking about a transnational company with very high level standards in software development and project management, such as CMMI, PMI certifications. So what is the problem? Why are people not adopting good practices?

We collect data from interviews with information architects, project leaders, project managers, Scrum masters, and Product owners, from national and international companies. The common agreement was: "at the end there is no need to document things", or "there is no time to do that", or "people is not aware of current software modeling standards and they just use what it works for them". Frequent and typical scenarios on the production line (where programmers are) they preferred to read a code rather than a design document to understand a project, nothing new, as pointed in [21].

So, we start tracking back the problem, to the University. Colleague students also do not like to design and do diagrams of their projects. In every single contest when you ask them about methodologies, models, design, and so on, they just simply reply: "we are agile, no need to document". From where they got this misconception of agile development? even worse, why the contest is not evaluating the quality of the product and just simply relying on five minutes prove of concept? Perhaps, in a hackathon this is a valid assumption due to the limited time they have to produce their projects, anyway not doing that means that normally they do not adopt a software engineering strategy before starting to code.

So we moved backward to the contest and start reviewing software engineering courses content, exams, projects, and surprisingly, everything students need to produce quality products based on software engineering methodologies was there. Unfortunately, few students remembered how to use them correctly, when and how?

### 3.1 Year One: Doing the Wrong Thing

At the university we have two courses, Research and Development I and II, in one course students are expected to do the initial phase and the planning of a big project, while in the second, the implementation, control and deployment. We have experimented with two groups of 30 students each, forming teams of 6 students during the first year. Originally, the role of the professor was planned to be a coach or mentor, as we expected

that senior students would have already all the skills and knowledge required to do their work. As this was not the case, then some training was needed to guide them in the analysis and design of their projects.

The first step was to define the problem. Techniques such as review of existing literature, participatory observation, interviews, activities diary, Market research, and FODA. Then, they were asked to formalize their findings using PERSONAS for the user modelling; requirements were specified with UML-use cases; and business process with either UML-activity diagrams, Petri Nets with Yawl or BPMN. From the students, 80% did not show any skill to do correctly the diagrams. The most common problems were the following:

- 90% of the students did not know the notations or the technique.
- 95% of the students did not use naming conventions correctly for the use cases and process. Instead of using a verb + subject, they just simply used a meaningless subject.
- 80% of the students did not use the complete notation, for instance, when modelling use cases: insert, include and inheritance relationships were missing. Even that during the review of this topic those relationships were mentioned to be important.
- 75% of students did not model correctly the business process. Most of the problems were related to the correct mapping of the actual process. Recursive tasks, parallel and splitting sequences were missing in most of the diagrams.

We assume the lack of practice on the design and modelling activities were one of the reasons why at the beginning students performed poorly. We use four iterations but results were just good with one out the six teams. Nevertheless, the rest of the teams had acceptable models. Thus confirming our first hypothesis, that first poor results were due to the missing experience modelling.

The second activity during the first course was planning. Techniques such as: COCOMO, Delphi, Planning Poker, and Function Points were proposed to students. To understand the complexity of their project they were asked to design: class diagrams, sequence diagrams and state diagrams, the data-base E-R, and the overall architecture using MVC abstractions. Similarly, as in the first phase, knowledge and experience on these topics were missing. And particularly, the quality of these diagrams was worse. The most frequent problems were:

- 100% had to learn a technique to estimate the size of their project.
- 85% of the students did not use correctly name conventions for class diagrams.
- 90% of the students did not use any pattern when modelling class diagrams. Simply simplifications and mandatory methods such as CRUD were missing.
- Just one student of the 60 knew how to write a MVC architecture for their project.
- None of the students knew about patterns.
- Sequence diagrams were the most complex topic and just one student did it right.

The final outcome from this course was non-functional prototypes, including the User Interface design, see Figure 1 with some examples. The common issue with this activity was that there was no standard graphical notation to present low-level UI prototypes.

The consequence of not having or adopting the same notation, or not using the notation correctly was that during the presentation of their projects there were confusion, constant disagreement which was normal, and not the best atmosphere to foster a positive interchange of perspectives among the different students.

During the second course the biggest problem was the assumption that waterfall methodology could be applied, thus, we expect students to simply implement the design proposed in the first course. Most of the problems were related to team organization, as members of the team were not the same from course one and two. Working on the same project was not a motivation for new team members. So, we discard those, new teams (4) in our study for the second half of the year. So, from the remaining teams (8) the problem was related to project management and the impossibility to follow the established plan. For different reasons but mostly:

- There was limited experience programming big projects that require collaboration of one or two members.
- Some students did not accept to share their code thus they prefer to work alone on the development and let their colleagues write the documentation.
- The plan predicted that people would work on a regular basis but this was not the case as students just shared a common space for 5 hours per week.

- Social networks were just useful to set up a common space for interaction but as communication was asynchronous most of the information was lost in the timeline.

The assumption that students were more motivated and self-organized during this course was just utopic. The truth is that they just simply fail as most teams just never found a way to work together. The best two teams succeed just to develop 60% of the plan. The rest just finished between 40 to 50% of the plan. Important characteristics, such as dashboards, that were planned were not developed. In another scenario, two web applications were designed, one for academic purposes and the second one was related to a health care electronic record system. Both projects had problems to conduct a quality strategy in the development as there were no technique to validate the quality is the product, i.e., the functionality correspond to the plan and user expectations were fully covered. The problems to finish on time the project were:

- Students were not familiarized to the user test, and conducting real qualitative or quantitative evaluations of their projects. Thus, at least two weeks of delay to learn about this topic.
- Interaction with end users was considered in the plan. In all the cases, there was a real beneficiary of the project, a school, an institution. So conduction research with real end-users was possible. Consequently, we had to devote some time to discuss with them.
- Common changes. It was the case on every single project when they were close to the end a lot of new requirements and changes emerged students did not take well that situation and a generated a lot of stress.

The final result of one year of work was on the benefit for the future of these courses. The lesson learned was on the benefit for the generation of students as discussed in the next section.

### 3.2 Year Two: on the Need for the Right Strategy

So the first thing we did was to reorganize both courses. Coaching was not anymore the only teaching strategy, just for those groups of students with a certain degree of knowledge. Second, a diagnosis test was created to identify their knowledge skills with regard to analysis and design of software, and product

development. Third, course content was planned to regularized students to guarantee common comprehension of the design and analysis of software. Fourth, drop the waterfall methodology and rely on an agile strategy. Fifth, starting from scratch a project was optional; students had access to the repository of existing projects (8) to understand the ultimate goal of the course and letting them choose any of those projects to expand them. Finally, reduce the discomfort of some students to share their projects code by writing a document to register the authorship of the software.

During the adopting of the new strategy, we found new problems:

- The quality of the diagrams did not improve anything compared to the previous year. The complexity of the notation, the lack of experience, and cognitive load while modeling different elements using different notations was reported to be a problem.
- Planning was not done correctly, most of the teams, 6 out of 8, rely on COCOMO technique but applied wrong, the rest used function points but wrongly as well.
- Architecture and pattern-based models were still missing.
- Evaluation of software was not part of the process again.
- Every project start from scratch, nobody was interested in previous projects. The consequence was that even for professors everything was new.
- Progress on the projects still ranges around 40 to 60% of the plan.
- Change in the methodology was not a good solution.

The last item was the most critics for the second year. Everybody wanted to agile but nobody, including professors running project adopting an agile strategy. Students misunderstood most of the principles of agile paradigm; the most frequent problem was related to documenting the design of their products. Everybody assumed that there was no need to document anything. Also, when it came the moment to test the software, students were reluctant to do that exhaustively as they assume that changing the code in the future would be easy so why bothering the developers with deep test. End users testing were let aside; nobody put much attention to end users. So, as a re-



sume, agile principles were not adopted so everybody obeys a free will strategy with not so good consequences. Nevertheless, the final product was closer to a quality product as all the previous misconceptions of an agile strategy were solved on time. At the end there was just a common problem related to the quality of the diagrams and the development, our challenge for the third year.

### 3.3 Year Three: Empowering Students

The third year, a professor was sent to do some training in SCRUM, with an international expert with a lot of real-life projects. Thus we change everything and stop doing a two-half project but a one-year project development adopting SCRUM as methodology, not just the principles but also the recommended techniques for each phase, in next section we describe what so far have proven to be the best methodology to develop research and development projects in the academic context. One of the most important aspects was to empower students, given them the freedom to choose but also guiding them to make decisions wisely. The most interesting aspect was related to diagrams and notations, how they change them and how good they are while discussing during the presentation of their projects.

## 4. A Methodology to Develop Research and Development Projects at University

These are the steps that after two years of work have probe to be the best techniques and notations for the development of R&D projects. The methodology under the current design process is analyzed and represented in accordance with ISO/IEC 24744 standard “Software Engineering Metamodel for Development Methodologies” [29]. This forthcoming standard allows describing development methodologies in terms of “Metamodel” and offers a standardized set of notations for the methodology formalization. Moreover, the main elements used for the formalization are described in Figure 1. These elements are graphical items that identify in a clear way a set of information like process steps, build tasks, task producers, techniques, tools, relationships.



Figure 1. The graphical notation of the ISO/IEC 24744 standard, source [29].

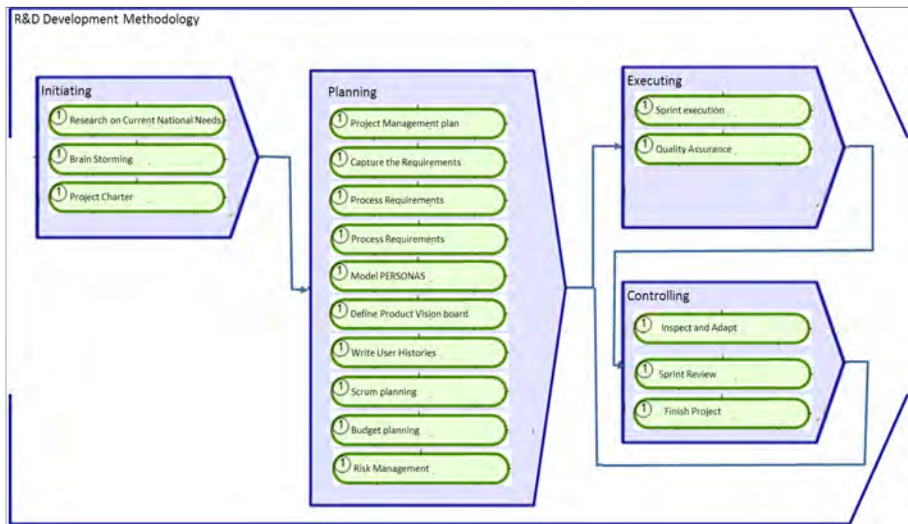


Figure 2. A methodology to develop R&D projects for novice engineers in Information Technology at the University, using the notation of the ISO/IEC 24744 standard [29].

In this chapter we do not pretend to present in details the resulting methodology just the current design process description, as it is shown in Figure 2. In Step 1, initiating. Students are invited to do some research current national needs, in Mexico they include: Education, health, security, human rights, or scientific development.

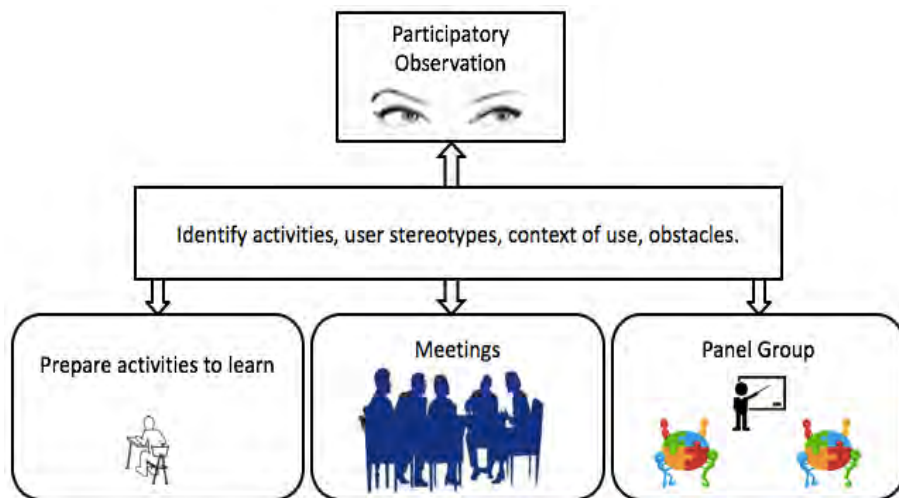


Figure 3. The process of gathering requirements

Once they got inspired in the nature of the problem, vital step to adopt an agile strategy, then they use the brain storming technique to decide what to do about the problem. Once a decision was made they had to write a project charter. Next Step is planning. The recommended techniques are related to project management planning: FODA and market analysis. Then “Capture Requirements”, to analyze the user needs (Customer requirements) and decompose them in a series of numbered, single-item requirements that are formally agreed and prioritized. The activities enable to verify the reception and understanding of the final users’ needs by the design team provide both that team and the final users with an agreed list of technical high-level requirements that will be bound to the development of the system. Selecting the right place where Participatory observation could be conducted, then observe. Once you get basic understanding of the problem, then writing the appropriate questionnaires to extend our comprehension and asking the stakeholders of the product to gather as much information as they know about how they currently do their work.

All these facts are relevant to create PERSONAS [31] and a Product Vision BOARD of the project [32]. To ensure that the system covers the needs of the context require producing chips with stereotypes PERSONAS [31]. This technique captures details HCI goals, desires, limitations, knowledge of end-users of an interactive system.

In short, the process of defining PERSONAS is to create user profiles. In principle, as a result of participant observation, interviews, and other activities listed above, we proceed to identify user profiles system. A PERSONA is a description of a group of users, typical of the system. Instead of talking about the group of users of an abstract, impersonal way a person plays a ‘reference’ of a user group and provides a means to talk and reason about this group through the characteristics of a fictitious individual, we create a character. The name, photo does not correspond to any actual person identified in our participant observation. The process of creating PEOPLE begins with the creation of fact sheets actors interviewed or observed, see Figure 4. Where each tab includes a type of information such as: skills, abilities, needs, desires, work habits, tasks and experience. These sheets of the observed actors should be categorized. Input, identifying actors and generating regions to group notes about them. For example, in a project for a school we identify: parents, teachers, school administrative staff and students. Then, for each observation groups to identify issues that could group each note. For example, for the teacher, we could create a group of notes related to “Provide challenges” to their students and this can be described in different ways.

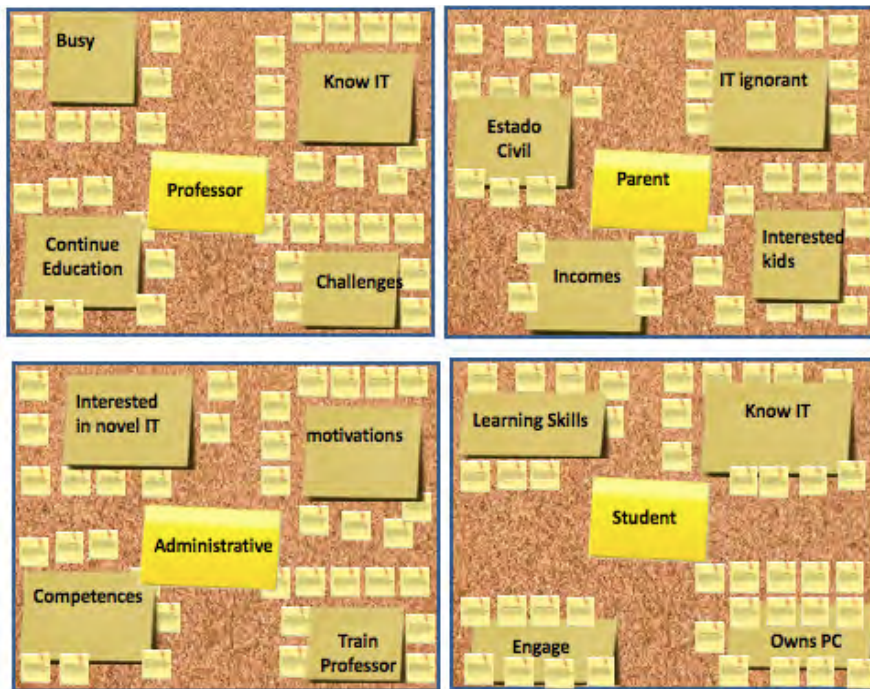


Figure 4. Peoples identified as a result of Participatory Observation

This method provides a significant constraint that has to do with the quality of the observations. That is, if we have lots of information as a result of the observations, which is very good, it is very likely that we have for each actor much associated with each data or even many groups of data information. Our proposal is to create a multidimensional map for each actor, where each group of concepts occupy a dimension. Let's review the example of actor student, see Figure 4. As part at least four groups of information are identified: learning skills, commitment, knowledge of IT, and availability of a PC at home. For each group of information one-dimension vector is created, where we need to identify the ranges of allowable values for that dimension is generated. Take note that these may vary between each set of information. Continuing the example in the context of technology-mediated learning we are interested only skill-based learning: visual, auditory, and kinesthetic. Then, learning skills can be grouped into three types, few (has only one skill), medium (has two skills), many (have the three developed skills). Similarly, we can categorize IT knowledge and commitment as short, medium, long. However, the category of availability of PC at home just requires analysis, with the following interpretation: little (no PC at home), medium (available PC at home but have to share use), much (available PC at home for personal use).

We must take into account that the values used for sizing each group, little, medium, high, are used illustratively as the names and their correspondences may be different, and not necessarily uniform in quantity. For example, the IT skills could be classified into four dimensions, not three, depending on the available information, as ignorant (neither knows nor uses TI), basic (knows some necessary IT classroom) bigot (use and is aware of new trends in IT and IT support used in the classroom), expert (IT develops solutions). The next step is to take up the sheets of our actors and classify each worksheet precisely at some level of each dimension, points around each value in the vectors of Figure 5.

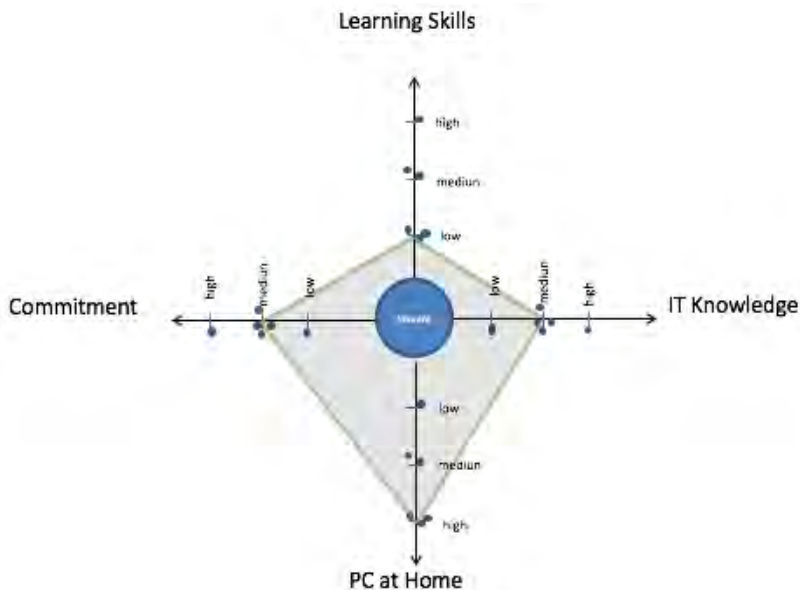


Figure 5. Multidimensional space

Since a PERSONA is actually fiction, include the actual data describing the important characteristics of a particular user group in a fictional character. So we need to ensure including relevant information where at least two data points around each value in the vectors of Figure 5. Derived from this process will emerge many proposals for PERSONAS, for our example, we have at least four for each actor, which results in sixteen actors, which is an excess of information. Since new research by Alan Baddeley [33] and Nelson Cowan [34] found that the human being is able to process 4 bits of information at a time [35]. And this is not only limited to the memory to the working memory but also to the long-term memory. It has been shown that humans can deal with information including category three things, if there are more things to remember for the ability to remember category down to 80% and so on up to 20%. This cognitive

limitation forces us to follow a selection process to reduce the 16 people to a number between 3 and 4.

This selection can be done formally by using space selection of options proposed by MacLean [36]. According to this method, a set of options (PERSONAS in our case) their usefulness or feasibility can be evaluated when considering the options as a question and the possible answers (priority criteria for our people) are evaluated. That is why the first step is the definition of relevant criteria for our person. Consistency with the project, easy to identify, a stereotype is intuitive: in this case to consider three criteria were identified. The weight assigned to each criterion is based on the experience of the designers of the people.

The significance of the links, figure 6, are solid darker line (++) means strong support, continuous dark line (+) means the support, the dotted line (-) means a neutral relationship, the dashed line (-) means little relationship, and the thick dashed line (-) means no relationship. Of the four candidates Students denoted in Figure 6, we place them as possible answers to the question “What is the best person for the stereotype of student?”.

	Opciones	Evaluación	Criterio
Which PERSONA is the best Student Stereotype?	Balanced Profile and High Availability	+	(1) Consistency with the project
		+	(2) Easy to Contact
		++	(3) Intuitive stereotype
	Balanced Profile and average availability	++	(1) Consistency with the project
		-	(2) Easy to Contact
		++	(3) Intuitive stereotype
	Balanced Profile and moderate availability	++	(1) Consistency with the project
		+	(2) Easy to Contact
		++	(3) Intuitive stereotype
	Balanced Profile	++	(1) Consistency with the project
		++	(2) Easy to Contact
		++	(3) Intuitive stereotype

Figure 6. Selecting Options Design Space, source [36]

We use a letter to identify each student PERSONA, the same as we see in Figure 6. In general, we can see, see Figure 6, which are all intuitive stereotype and generally all are strongly consistent with the project. However, the variable that helps us determine who

opted has to do with the availability to identify these users. Note that this analysis only provides an overview of the different properties related to the person and not reach a conclusion about what is the best representation. It is simply an aid to select an option among several in a structured way. This technique can serve for decision-making at all stages of a project and even our life.

The motivation for the team is important and by doing this activity at least we guarantee that the students' team understand the need of the organization, the value that doing the product would add to the stakeholders, thus never losing the motivation in our students. Writing user histories instead of use cases is the next step. Then, writing the Tests for each user history and adopt a test driven development strategy.

During the execution, an introduction to boundary test and equivalent partition techniques was key at this stage. Students became aware on the need that any line of code must have a corresponding test before it was implemented. Unit test was used in each project during the third year, thus writing tests was a common activity for the students, and thus, they were closer to a better quality product. Usability studies were also recommended, by using formal studies such as the IBM Test suite. The advantage of the initial stage and the connection with end-users since the beginning simplify the evaluation process and reinforce the in the perspective of the students the need to produce the expected product just as the final-users needed. Then again we had the problem of doing a reverse engineering process on the running artifacts and writing quality documentations with the findings reported in next section.

Finally, each sprint was controlled with its review. Inspect and adapt technique were used as well to improve performance in next sprints. Iterations were made as long as the time allows, final solutions we presented in a workshop where every team got feedback from their colleagues. Each team is invited to present their work in less than 20 minutes, focusing on technical challenges to foster discussion among their colleagues.

## **5. Case Study: The Development of a Serious Game to Teach Kids about Cancer**

As an example of the resulting methodology, we briefly show a real life problem, just to illustrate the kind of research our students do. They were asked to research on a public health problem and how virtual reality could be used as a solution to any of those problems. The use of Virtual Reality was a constraint defined by the team, a serious game to help kids and relatives to understand Cancer treatment. Being a growing problem in



Mexico, Cancer normally affects patient's perspective about their future, badly when you are grown up, but this is not the case for a kid. Even kids with Cancer are not affected by bad news but adults are normally reacting differently. So, our first goal of the serious game was to help adults to understand the disease. The second, to help kids to understand the treatment, particularly chemotherapy as kids are more afraid of needle than Cancer itself.

During the planning of the project, they started to work with an association from where they did the requirements capture, using techniques such as participatory observation, interviews, activities diary, video and audio recording, focus group. Decided to focus only childhood cancer, they confirmed the problem and the fear that children feel towards treatments, they know what they are going to do them or what will happen. Also, the fact of not knowing what the disease itself, that causes stress in children every time they receive their treatment to the point that they leave many treatments, and the consequences are not just healthy but economical.

The challenge, how to create a solution and in which context the solution should work. Children suffering from cancer have the following requirements:

- Emotional needs that are common to all children, even when they are. It is including the need to feel loved, to develop a sense of belonging, to feel self-respect, to get a sense of accomplishment, security and self-knowledge, and be free of guilt.
- Needs arising from the child's reaction to the disease, hospitalization and treatment. Fears and feelings of guilt or worthlessness require support, love, empathy, understanding, approval, friendship, safety, compassion and discipline.
- Needs arising from the conception that the child is going to die, which generates reactions of fear, anxiety, loneliness, sadness.

Pain is a mechanism that warns us of the existence of an injury or physical illness. In patients with cancer pain, it is very common and disrupts their quality of life. Although usually not fully relieved in some patients, most of them the pain can be controlled effectively using non-pharmacological treatments. Highlights distracting techniques, the use of imagination and training in relaxation / breathing, therapeutic elements considered essential to relieve pain and reduce anxiety.

Distraction is a cognitive technique that involves drawing the attention of the source of pain and directs it to other types of stimulation. With cancer patients it has been used



external distractions such as video games, toys, dolls or stories, and internal distractions and emotional images. Distraction is involved in various psychological procedures or because the child is immersed in the fantasy, because it must cater to the rhythm of your breathing or that focuses attention on an attractive activity.

The proposed solution is to design a serious game in which children receive their first radiotherapy or chemotherapy, so when they can assimilate the advantages of the treatment, and even consequences of not getting them. The game is divided into different interactive systems, one part consists of multimedia content describing the disease to every person in the waiting room, and this is inspired in techniques to manipulate crowds.

Then, the selection of the metaphor for the game, healthy human cells are presented as maize. That is affected by Cancer cells, maize smut in our metaphor that expands as fast as possible unless you got a cure. As a treatment, the game is divided into stages, a main game in which the path to be followed in the treatment develops, and different mini games related to this first. While the patient is receiving chemotherapy or radiation therapy is to divert attention from what is really happening and also to cope with the long periods in which it applies.

The support material is also an important part, whether with videos, animations, stories, or any material that may be of assistance will be beneficial with this information is intended for patients of what the disease is, as develops, different types, the effects it causes, as well as relevant to treatment. The use of metaphors will be vital since we cannot treat the medical terms to explain to a young child, which is why we have been raised using series of metaphors that help the patient to assimilate all the concepts without losing sight of the real meaning, for this it had to choose between different elements found in our daily lives, and could quickly figure corn is one of the metaphors we propose to introduce, as it is an element which we know from an early age, in addition to being available for the entire population.

Since the human body can have harmful microorganisms, corn presents a series of visible fungi shown to infect. He thought of a corn, because in Mexico it is an element that much of the population knows and is familiar, together that is a non-aggressive figure which displays symbolically as an infection develops in a body easily, so that children understand each part of the process.

Representing a healthy corn and other infected. In addition to the corn themed main theme we propose a farm which represents steps (barns) that must be followed during

the treatment, each game barn with a special informative content. Activities that were raised were created from the goals that have medical procedures: The patient must take medication, as well as the processes of radiation and chemotherapy, so he thought of an activity in which the objective is to collect pills to improve the current condition of the patient.



Figure 5. The farmer as metaphor of a doctor taking care of the sick corn (kid with cancer)

Tests were conducted with children between 6-10 years of the center “A New Hope” for treating children with cancer. Including those dedicated activities for each point of the system they were tested. As a first impression sympathy of children was obtained by knowing who would play instead of waiting. On a scale of 1 to 10 enthusiasm of 10 being the highest number and children were categorized expected to average 8-9 evaluated by psychologists from the center. When they were previously cataloged by an average of 6.8, with an indifferent state or seriously did the children receive chemotherapy.

As for the activities children’s able to play properly after a brief explanation of how they had to use, although a percentage of children showed some coordination problems, so it is recommended to have certain levels in applications because it does not all children have the same cognitive and motor capacity.

The results indicate that most of the surveyed users found the system useful, simple and with a real purpose, which serves as a tool for treatment by traversing children. The conclusion, these activities were a major influence on the behavior of children since improved his mood. This gave us pause to think that there are some groups of people, of which certain aspects have been neglected, in this case, child with cancer in our society. Future work is more activities and generates a metric to know how effective the system for them.

## 6. Discussion

During the three years, we conducted research around 30 research projects were developed, around ten per year. As mentioned before, we moved from a waterfall development methodology and project management to an agile project development strategy. The last prove to be the best one in terms of quality of the diagrams. At least, from our interviews some of the reasons were the following:

- Lack if time to rework on diagrams were assume correctly in previous steps. This was primarily the case when adopting the waterfall methodology. Most of the students used UML diagrams: activity, use cases, class, sequence and state-charts. It was really stressing to students to re-do their work. So 100% of the diagrams did not correspond to the final solution. Consequently, they were useless.
- On the utility of the diagrams. This was the common claim. Working software rather than exhaustive documentation. This common misunderstanding of agile strategies was a problem during year three. However, we had more acceptable results. 100% of the diagrams corresponded to the results of accepted in the previous Sprint. This was clearly a huge improvement in the quality of the content of the diagrams.
- On the correct use of the notation. Constantly iterating the review of the diagrams improves the correct use of notations. Inexperienced developers need constant feedback and the final result was clearly better. In year one and two, they just did UML diagrams. In year three, it was open the use of diagrams. The use of BPMN, Task models and other notations was allowed, even colored UML. The correctness of the diagrams dramatically improved. In year one the problem was not just about the correlation but also about the correct use of notations. Common mistakes were the use of the wrong diagrams for a specific model, most commonly a sequence diagram to represent business process.
- On the improvement/modification of existing notations. Giving freedom to students to modify or improve existing notations showed to be reality useful to foster discussion among them and to improve the quality and abstraction of the models.

Our work, found common problems already reported in the literature. Modify the Notation but Rely on existing guidelines. If any modification is going to be made, consider the work of Jacques Bertin's Semiology of Graphics [8]. Use the full capacity of visual variables [5], see Figure 6. The choice of visual variables has a major impact on cognitive effectiveness as it affects both speed and accuracy of interpretation [3,12, 20].









PLANAR VARIABLES		RETINAL VARIABLES		
Horizontal Position (x)		Shape	Color	Size
				
Vertical Position (y)		Brightness	Orientation	Texture
				

Figure 6. Guidelines to develop a graphical notation. Source [8] in [5].

Sequence Diagrams are an important model that communicates the order of ejection of the processes in a system. Diagrams considered really important and not producing them with enough quality normally derives errors in the final product [1]. Adding visual aids proved to be useful as shown in Figure 7. Visual CUES [9] indicating which elements in a diagram are related to real objects.

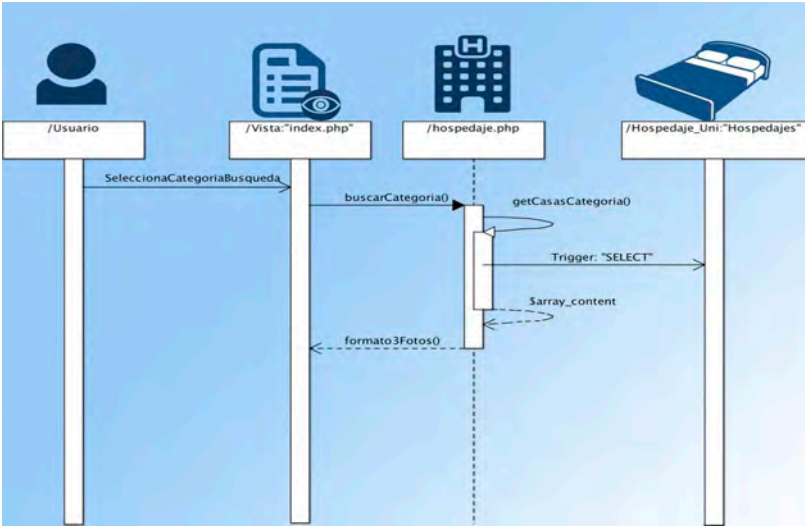


Figure 7. Visual AIDS to improve comprehension of Diagrams, in this example in sequence diagrams is improved with a graphical image depicting the object. Below color schema added to the navigation map of the application.

Shapes are important to understand the difference of non-similar objects [3], even that this research was conducted mainly for data graphs, their findings prove that

people tend to understand better when we use simple but different shape objects and differentiate them with size. In UML classes are sized differently based on their content but this do not mean anything about their complexity, yet students always discuss about implementing the smallest class, just by simply relying on their visual perception.

Ask Students. As pointed in different context. Sometimes when it is not clear the notation you can ask students to propose their own. As reported in [5]. In our case, we asked our students to produce their own diagrams, even UI prototypes widgets.

## 7. Conclusion

In this chapter we present a serious of findings based on three years of research while teaching novice software engineers, senior students of the university, how to produce a quality product. While the whole study focused in the whole process, we wanted to make emphasis in this chapter on the complexity and poor quality of the documentation. Our findings confirm what it is written in the literature, several are the causes why diagrams are not easy encoded or understood. Our research proved that by allowing some strategies recommended in the literature, mostly by empowering students in selecting the guideline that fits better to their needs produce better documentation. Still, there is a need to keep working on this topic and still is an open issue, as we cannot imagine a scenario where each time we work on a project we could promote the adoption of a modified notation, thus losing the benefits of adopting a standard. Perhaps, in the future we would join efforts towards an easier to understand notation.

## 8. Acknowledgements

To students of the computer science faculty of the Meritorious Autonomous University of Puebla that were part of this research.

## 9. References

- [1] Bolloju, N., & Leung, F. S. (2006). Assisting novice analysts in developing quality conceptual models with UML. *Communications of the ACM*, 49(7), 108-112.
- [2] Davis, C. J., & Hevner, A. R. (2015). Neurophysiological Analysis of Visual Syntax in Design. In *Information Systems and Neuroscience* (pp. 99-105). Springer International Publishing.
- [3] Cleveland, W.S., McGill, R.: Graphical perception: theory, experimentation, and application to the development of graphical methods. *J. Am. Stat. Assoc.* 79(387), 531-554 (1984).

- [4] Coad, P., Luca, J. D., & Lefebvre, E. (1999). *Java Modeling Color with Uml: Enterprise Components and Process with Cdrom*. Prentice Hall PTR.
- [5] El Kouhen, A., Gherbi, A., Dumoulin, C., & Khendek, F. (2015). On the Semantic Transparency of Visual Notations: Experiments with UML. In *SDL 2015: Model-Driven Engineering for Smart Cities* (pp. 122-137). Springer International Publishing.
- [6] Figl, K., Mendling, J., & Strembeck, M. (2013). The influence of notational deficiencies on process model comprehension. *Journal of the Association for Information Systems*, 14(6), 312.
- [7] Genon, N., Heymans, P., & Amyot, D. (2010). Analyzing the cognitive effectiveness of the BPMN 2.0 visual notation. In *Software Language Engineering* (pp. 377-396). Springer Berlin Heidelberg.
- [8] Jacques, B.: *Semiology of Graphics: Diagrams, Networks, Maps*. University of Wisconsin Press, Madison, Wisconsin (1983).
- [9] Kim, J., Hahn, J., & Hahn, H. (2000). How do we understand a system with (so) many diagrams? Cognitive integration processes in diagrammatic reasoning. *Information Systems Research*, 11(3), 284-303.
- [10] Koschke, R. (2003). Software visualization in software maintenance, reverse engineering, and re-engineering: a research survey. *Journal of Software Maintenance and Evolution: Research and Practice*, 15(2), 87-109.
- [11] Kutar, M., Britton, C., & Barker, T. (2002). A comparison of empirical study and cognitive dimensions' analysis in the evaluation of UML diagrams. In *Procs of the 14th Workshop of the Psychology of Programming Interest Group (PPIG 14)*.
- [12] Lohse, G.L.: A cognitive model for understanding graphical perception. *Human-Computer Interaction* 8(4), 353-388 (1993)
- [14] Moody, D., & van Hilleghersberg, J. (2008). Evaluating the visual syntax of UML: An analysis of the cognitive effectiveness of the UML family of diagrams. In *Software Language Engineering* (pp. 16-34). Springer Berlin Heidelberg.
- [15] Moody, D. L., Heymans, P., & Matulevicius, R. (2009, August). Improving the effectiveness of visual representations in requirements engineering: An evaluation of i\* visual syntax. In *Requirements Engineering Conference, 2009. RE'09. 17th IEEE International* (pp. 171-180).
- [16] Moody, D. L. (2009). The "physics" of notations: toward a scientific basis for constructing visual notations in software engineering. *Software Engineering, IEEE Transactions on*, 35(6), 756-779.
- [17] Siau, K., & Loo, P. P. (2006). Identifying difficulties in learning UML. *Information Systems Management*, 23(3), 43-51.
- [18] Rosa, M. L., Ter Hofstede, A. H., Wohed, P., Reijers, H. A., Mendling, J., & Van der Aalst, W. M. (2011). Managing process model complexity via concrete syntax modifications. *Industrial Informatics, IEEE Transactions on*, 7(2), 255-265.

- [19] Stanculescu, A., Vanderdonckt, J., & Mens, T. (2008, May). Colored graph transformation rules for model-driven engineering of multi-target systems. In *Proceedings of the third international workshop on Graph and model transformations* (pp. 37-44). ACM.
- [20] Winn, W.: An account of how readers search for information in diagrams. *Contemp. Educ. Psychol.* 18(2), 162–185 (1993).
- [21] Scanniello, G., Gravino, C., Genero, M., Cruz-Lemus, J. A., & Tortora, G. (2014). On the impact of UML analysis models on source-code comprehensibility and modifiability. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 23(2), 13.
- [22] Cowan, Nelson. 2001. "The magical number 4 in short-term memory: A reconsideration of mental storage capacity." *Behavioral and Brain Sciences* 24: 87–185.
- [23] Odell, J., Parunak, H. V. D., & Bauer, B. (2000). Extending UML for agents. *Ann Arbor*, 1001, 48103.
- [24] Baumeister, H., Koch, N., & Mandel, L. (1999). Towards a UML extension for hypermedia design. In «UML»'99—The Unified Modeling Language (pp. 614-629). Springer Berlin Heidelberg.
- [25] Rodríguez, A., Fernández-Medina, E., & Piattini, M. (2007). A BPMN extension for the modeling of security requirements in business processes. *IEICE transactions on information and systems*, 90(4), 745-752.
- [26] Van der Aalst, W. M. (1993). Interval timed coloured Petri nets and their analysis (pp. 453-472). Springer Berlin Heidelberg.
- [27] Guerrero-García, J. (2014). Evolutionary design of user interfaces for workflow information systems. *Science of Computer Programming*, 86, 89-102.
- [28] Fonseca, J. M. C., Calleros, J. M. G., Meixner, G., Paterno, F., Pullmann, J., Raggett, D., ... & Vanderdonckt, J. (2010). Model-based ui xg final report. W3C Incubator Group Report, May, 32.
- [29] ISO/IEC: 24744: Software Engineering - Metamodel for Development Methodologies. International Organization for Standardization/International Electrotechnical Commission, Geneva, 2007.
- [30] Guerrero-García, J., Gonzalez-Calleros, J., Muñoz-Arteaga, J., Morales, A., & Monarca, I. (2016, April). Getting Research Findings into Practice: Guidelines to Produce Quality Software Engineering Diagrams to Assist Novice Engineers. In *2016 4th International Conference in Software Engineering Research and Innovation (CONISOFT)* (pp. 149-158). IEEE.
- [31] Pruitt, J., Adlin, T., *The Persona Lifecycle: Keeping People in Mind Throughout Product Design*. Morgan Kaufmann, 2006.
- [32] Pichler R. (2016). *Strategize: Product Strategy and Product Roadmap Practices for the Digital Age*. Pichler Consulting, 1 edition.

- [33] Baddeley, Alan D. 1994. "The magical number seven: Still magic after all these years?" *Psychological Review* 101: 353–6.
- [34] Cowan, Nelson. 2001. "The magical number 4 in short-term memory: A reconsideration of mental storage capacity." *Behavioral and Brain Sciences* 24: 87–185.
- [35] Veenma, M., P. Wilhelm: The relation between intellectual and meta-cognitive skills from a developmental perspective, 14, 89-109 *Learning and Instruction*, 2004.
- [36] MacLean, A., Young, R.M., Bellotti, V., Moran, T.P., Questions, Options, and Criteria: Elements of Design Space Analysis, *Human-Computer Interaction*, Vol. 6, No. 3-4, 1991, pp. 201–250.



# Chapter # 7

## Improving Privacy Notices Usability Applying Cognitive Ergonomics in Interaction Patterns

Enrique Sánchez Lara  
Universidad Popular  
Autónoma del Estado  
de Puebla  
Puebla, Puebla 72410,  
México  
enrique.sanchez@  
upaep.mx

Sandra R. Murillo  
Universidad Popular  
Autónoma del Estado  
de Puebla  
Puebla, Puebla 72410,  
México  
sandrarcio.murillo@  
upaep.mx

J. Alfredo Sánchez  
Universidad de las  
Américas Puebla  
Cholula, Puebla 72810,  
México  
alfredo.sanchez@  
udlap.mx

### 1. Introduction

Internet services, cloud computing, network applications and social networks have changed the way people interact and work today. In many cases, personal information is requested in order to access them and the end user is not aware about where that information will be stored, how it is or not protected and when and how it could be used again with or without his explicit consent. From the perspective of personal data protection, owners of data have the right and freedom to decide what to communicate, when and to whom, maintaining control over their personal information at all times. Due to problems that have arisen (identity theft, fraud, etc.), international organizations have proposed mechanisms for software developers that include options that strengthen user privacy. This paper investigates what tools developers use to produce privacy software and propose an interaction pattern in order to enhance user experience with the elements of a privacy notice. The paper is organized as follows: Section II discusses the main information privacy and cognitive ergonomics issues. The general and specific objectives are defined in Section III. Section IV describes the current situation as regards privacy policies. Legacy tools versus interaction patterns are presented in Section V. Finally, Section VI presents the conclusions of this study.

## 2. Privacy information and cognitive ergonomics

Protecting information began in the world focused in protecting national security and government information, for example the Administrative Procedure Act of the United States in 1946 and the Privacy Act in 1974. As time have passed and technology have evolved, new kind of laws have been necessary, including the protection of people's information and privacy [1]. Since 1967 in Europe with Resolution 509, in the 70's in France, Germany and other countries, later on with the promulgation of Convention number 108 to protect people against automated data processing the world began to focus in people's right to privacy. In 1995, Directive number 95/46/CE of the European Council chapters 7-8 discusses the differences between the governments of the European Community about the protection of personal privacy and proposes that there should be a common base of user privacy protection rights eliminating existing disparities.

These provisions established users' right to access their information and modify it when inaccurate, their right to cancel data when it is no longer relevant or has become obsolete, and their right to oppose their use by a third party. Several projects have been implemented in order to achieve or enhance computer user privacy and computer user privacy notices [2] [3] [4] [5] [6]; for example, in the United States the P3P protocol was promoted by Microsoft in an attempt that Web pages would create an internal structure that assured some degree of user privacy [2]. However, experiences in Latin American environments have not been considered in those developments and proposals.

In Mexico, the Federal Law on Protection of Personal Data Held by Private Parties (LFPDPPP) [7] was adopted in 2010. According to Article 16 of this law, protecting personal data is a fundamental right. Any institution or enterprise requesting personal information should create and publish a Privacy Notice, a document that explains some or all of the ways the institution or enterprise gathers, uses, discloses and manages a client's data. This privacy notice should specify the enforcement, rectification, cancellation and opposition of access rights. Any company or institution must consult owners as to whether they authorize the transfer of their information to third parties. The institution must notify owners about any changes made in the Privacy Notice and request for consent to use new data. The Privacy Notice should be made available through physical, digital, visual, audio or any other media. All individuals or their legal representatives may exercise any of these rights, without cost. The LFPDPPP specifies three possible modes of the Privacy Notice as follows [7]:

### Full Privacy Notice (Indispensable)

1. Identity and address of the person or entity responsible of gathering personal data.
2. Purpose of the personal data processing
3. Options for limiting the use and disclosure of personal data
4. Options to exercise ARCO rights
5. Procedure for communicating changes in the Privacy Notice to the owner of personal data.
6. Where appropriate, data transfers that are made
7. Whether they are managing sensitive personal data

### Simplified Privacy Notice

1. Identity and address of the person or entity responsible of gathering personal data
2. Purpose of the personal data processing
3. Mechanisms to find the full Privacy Notice

### Short Privacy Notice

1. Identity and address of the person or entity responsible of gathering personal data
2. Purpose of data processing, with no need to identify secondary purposes
3. Mechanisms by which owners may find the full Privacy Notice

There are several guidelines, interaction patterns [8], tools [9] [10] [11] and research projects [12] that have proposed and evaluated [13] mechanisms to generate a privacy notice. This facilitates the task of creating it, fulfilling the requirements indicated by law. However, the information generated does not produce a useful document for end users, neither enhances the user experience or cognitive ergonomics of the computer interface, delegating responsibility to the default options suggested by the interface [14].

Because current privacy notices are not standard, end users can't apply the experience gained consulting one when consulting a new one. Legal terminology and the poor level of information security culture produces uncertainty, bothersome, and a bad end user experience and some degree of mental fatigue because of the low level of cognitive ergonomics in the computer interface. Due to this situation, it is not easy

to recognize and exercise end user rights in this area even when the website they have accessed provides technical mechanisms for doing so [14]. None of the existing papers or projects proposes a detailed plan to represent, create and deploy user friendly privacy notices.

According to the International Ergonomics Association [15], “Ergonomics (or human factors) is the scientific discipline concerned with the understanding of interactions among humans and other elements of a system, and the profession that applies theory, principles, data and methods to design in order to optimize human well-being and overall system performance”. So, in this broad definition humans can perform better in an ergonomic environment. In the Human Computer Interaction arena, a more specific definition is necessary: Cognitive Ergonomics, that is defined in [15] as: “a discipline concerned with mental processes, such as perception, memory, reasoning, and motor response, as they affect interactions among humans and other elements of a system. (Relevant topics include mental workload, decision-making, skilled performance, human-computer interaction, human reliability, work stress and training as these may relate to human-system design.). In order to provide an ergonomic design to the interaction patterns of this work, as well as a very simple and easy to understand classification of the attributes of the privacy notice, the theories of [16] [17] were applied. In [16] the concepts of visual thinking states that the human brain is capable of processing in a natural and parallel way the following items:

- What
- Who
- How many
- Where
- When
- How

The reason of such six element’s parallel processing capability seems to be the need for surviving. Humans developed such parallel processing because, in a threatening environment it is necessary to know: What or Who is the enemy of threat, how many, where the enemy is, when the danger is present and how to deal with all of these elements and take a decision. In *Phantoms in the Brain* [17], the neurobiological fundamentals of such parallel processing are studied and discussed. For example, it is described that the human senses have separate areas in the brain to do their specific job, and everyone has a separate working memory area, allowing people to work in parallel in order to make almost instant decisions.

So, the specific attributes of the privacy notice and the particular elements of the end user interface were framed inside these global items (What-Who, How many, Where, When, How) in order to produce the final interaction pattern which is expected to cause less mental fatigue and avoid the end user mental workload limit [17]. The correlations between the particular end user interface elements were measured with a parametric analysis using frequencies. The results are described in Section V.

### 3. Objective

The goal of this research is to evaluate the use of interaction patterns with cognitive ergonomics elements in building.

#### 3.1 Specific objectives

1. To identify the current structure of the privacy notice on Internet pages.
2. To identify human factors interacting with privacy notices.
3. To propose a mechanism that allows developers to build effective management of user privacy applications.

#### 3.2 Methodology

A quantitative analysis of data was performed to examine the current situation as to online privacy notices in Mexico. It investigates whether they:

- » Have the elements required by law.
- » If users perceive a user-friendly format.
- » Measures the time invested in identifying its elements.
- » Documents user satisfaction related to this interaction with the computer interface.

Online privacy notices and paper-based questionnaires were used to collect data. Subsequently, the performance of a group of developers who used legacy tools to build interacting privacy policy applications was documented and compared to determine whether they had an interaction pattern set in place from the beginning of the project. This research was performed with the participation of Information Technology students in a private university in the city of Puebla, Mexico. Their participation was voluntary and compensation was not offered.

## 4. The current situation as to privacy policies

The average time users dedicated to identify privacy notice elements is documented, as well as human factors as regards interaction with them.

### 4.1 Elements of privacy policy

An experiment was conducted in order to record the time spent to identify elements of the privacy notice.

#### 4.1.1 Procedure to determine current situation

During the Fall term of 2012, two ninth-semester Computer and Systems Engineering students from a university in the city of Puebla, Mexico, consulted fifteen randomly selected Mexican websites in the months of November and December of that year. In the first session, the role of a privacy notice and its components were explained to them. They worked one hour daily with a five-minute break between the readings of each of the fifteen privacy notices of the corresponding fifteen web pages.

Each inquiry of the experiment was made directly via computer and participants were asked to find each element of the privacy notice and report it as existing in a predefined electronic format. They were allowed to make additional notes on paper. The time spent by each participant is shown in Table 1.

#### 4.1.2 Results

- » In Table 1 we show that the average time needed to read a privacy notice was 10.3 minutes; a college level of reading and understanding was assumed.
- » In Table 2 we illustrate that no single privacy notice showed all the elements required by law in the fifteen corresponding web pages.
- » Table 2 shows that required items –Identity and Address of the person or entity responsible of gathering personal data, Purpose of data processing and Options to exercise ARCO rights– were found in 86.7% of the total sample.
- » Table 2 shows that the mandatory element Method and options by which changes are reported about the Privacy Notice was found in 80% of the total sample.
- » Finally, Table 2 shows that options for limiting the use and disclosure of personal data were found in seven out of the fifteen privacy notices in the corresponding web pages.

**Table 1. Time used to identify the essential elements in each privacy notice.**

Studied Site .MX Domain	Person A (min.)	Person B (min.)
Telmex.com	21	17
Esmas.com	30	20
Seccionamarilla.com	20	17
Tvazteca.com	9	7
Telcel.com	11	10
Iusacell.com	10	7
Unefon.com	7	5
Mecagable.com	8	10
Hospitalpuebla.com	7	6
Sanatorio.com	5	5
Hospitalbetania.com	12	8
Beneficienciaespanola.com	10	7
Upaep	5	4
Itesm	8	7
Cemexmexico.com	9	7

**Table 2. Presence of elements in privacy notices**

Privacy policy element	% of Presence in privacy notices
Identity and address of the person or entity responsible for gathering personal data	86.7%
Data processing purpose	86.7%
Options for limiting the use and disclosure of personal data	46.7%
Options to exercise ARCO rights	86.7%
Where applicable, personal data transfers that are made by the entity	73.3%
Means by which changes in the privacy notice are reported to the owner of personal data	80%
Where applicable, whether sensitive personal data is handled	33.3%

## 4.2 Human factors

A statistical study based on frequency analysis [18] was performed based on the previous experience of participants, using the third version of a paper-based questionnaire to document human factors related to reading a privacy notice.

### 4.2.1 Methodology

A quantitative data analysis was performed, based on a parametric analysis using frequencies. A questionnaire was applied to 36 eighth and ninth- semester Information Technology students. There were 24 men and 12 women. Participants were asked to rate five statements (listed and labeled from “a” to “e” in the next paragraphs) based on the following Likert scale:

- Strongly Disagree
- Disagree
- Neither Agree or Disagree
- Agree
- Strongly Agree

Study question: “In relation to Privacy Notices on the Internet and the privacy of my data: \_\_\_\_”

- It is very easy to identify the elements of a privacy notice
- Reading a privacy notice is enjoyable
- I perceive that the websites are concerned about helping me to manage my privacy. The general format of a privacy notice is similar in all pages I visit
- I think that any internet user can understand the contents of a privacy notice

### 4.2.2 Results

- In Table 3 to 6 we present no participants indicated “agree” or “strongly agree” with respect to the ease of identifying the elements of a privacy notice, the enjoyable nature of the task, the concern websites have about privacy, or the similarity of notice formats.
- In Table 3 and Table 4 we show 66.7% strongly disagree with the statements that it is very easy to identify the elements of a privacy notice and that reading of the privacy notice is enjoyable..
- In Table 5 we exhibit 33.3% strongly disagree with the statement that websites help manage end user privacy.
- In Table 6 we show 50% strongly disagree and 36.1% disagree with the consistency of privacy notice formats across all the web pages they visit.
- In Table 7 we exhibit with respect to whether they consider that any Internet user can understand the contents of a privacy notice: 16.7% strongly disagree, 50% disagree, 13.9% neither agree nor disagree, 16.7% agree and 2.8% strongly agree.



### 4.3 Discussion

This study was conducted with volunteer eighth- and ninth-semester Information Technology students who were not offered any rewards for their time. An advanced reading level was assumed. For future studies, it is recommended that other profiles be included and that the sample is larger. In some of the following tables Likert values (4 = Agree , 5 = Strongly Agree) were not included because the frequency was 0 all the times, so the percentage is 0.

**Table 3. It is very easy to identify the elements of a privacy notice**

Likert value	Frequency	Percentage
1	24	66.7
2	9	25.0
3	3	8.3
Total	36	100.0

**Table 4. Reading a privacy notice is enjoyable**

Likert value	Frequency	Percentage
1	24	66.7
2	7	19.4
3	5	13.9
Total	36	100.0

**Table 5. Perceive that the websites are concerned with helping manage privacy**

Likert value	Frequency	Percentage
1	12	33.3
2	8	22.2
3	16	44.4
Total	36	100.0

**Table 6. The general format of a privacy notice is consistent across all pages**

Likert value	Frequency	Percentage
1	18	50.0
2	13	36.1
3	5	13.9
Total	36	100.0

**Table 7. Any internet user can understand the contents of a privacy notice**

Likert value	Frequency	Percentage
1	6	16.7
2	18	50.0
3	5	13.9
4	6	16.7
5	1	2.8
Total	36	100.0

In order to protect personal data of end users, the presence of a privacy notice is required; however, these notices do not have a homogeneous structure as we can see in Table 6, nor do they include patterns, icons or universal formats to help end users understand their basic elements. Reading them generates boredom, uncertainty and/or indifference and this is a concern, because information security culture is at risk. In Table 7 we exhibit participants believe that most people do not understand the contents of a privacy notice. If this happens to people in higher education, it is essential to investigate what happens in other sectors of the population.

As shown in Table 1, it takes too long to read the average privacy notice, considering that this step is mandatory in order to access a web page that collects personal data. In Table 3, Table 4 and Table 5 shows for most users that it is not an enjoyable activity and they perceived that there were no mechanisms to help them manage their privacy.

If there were a standard format for presenting information, it could be easier to create privacy notices and users could identify each element in a simple and timely manner. Users are required to learn computer security terms that are not defined in common language and may have different meanings when used by developers, causing confusion. Based on these results, two suggestions can be made. First, the relationship between user interface and privacy human factors should be studied in order to propose schemes that enhance the browsing experience. In addition, an analysis should be made to determine whether privacy notice developers have the necessary tools at their disposal to facilitate this task.

## **5. Legacy vs Interaction Patterns Tools**

In this part of the project, the performance of developers in building an end-user privacy application using legacy tools versus using interaction patterns is compared [8] [19].

## 5.1 Specific objectives

- To identify tools used by developers to build applications related to end-user privacy functions.
- To evaluate the usefulness for developers of the tools mentioned above.
- To evaluate the relevance of a set of interaction patterns related to end user privacy for the construction of computer interfaces.

## 5.2 Methodology

The functional requirements of five applications involving end-user information privacy were defined. During Fall 2012 a study of a course group of eleven seventh-semester Computer Engineering and Software Engineering students was conducted in order to identify the software elements used to build privacy applications. The students were allowed to select the tools they considered appropriate for their design and implementation. After completing the course and evaluating the functionality of their applications, students were provided with a set of interaction patterns (COPEMMA [19] and the Privacy Notice which will be described in Section V.1) and each team was asked to estimate their impact on the finished work, if they had received such patterns at the beginning of the course.

Students participated under the premise that only compliance with the requested functional requirements would be assessed. Development teams did not have the help of legal experts, graphic designers, or similar professionals. Functional requirements were presented to each team without mentioning the purpose of this study. Each team selected its software development methodology, programming language, icons, images and organization of information on the screen. To obtain their concluding remarks, they were invited to participate voluntarily without any remuneration.

## 5.3 Study projects

There were five projects involved in this study. These projects were presented at the exhibition of final projects organized by the Information Technology Department to which the students belong. In all five cases, evaluators and other students who participated in the project commented that these tools made it easier to understand privacy policies. The projects are as follows:

- In Figure 1 we show the navigation prototype for a Privacy Notice on a mobile device based on Android.

- In Figure 2 we show how web graphics display elements of a Privacy Notice for college students.
- In Figure 3 we show a simplified Android Privacy Notice based on learning styles.
- In Figure 4 we present Augmented Reality applied to display elements of privacy.
- In Figure 5 we exhibit a Display privacy policies based on P3P (Platform for Privacy Preferences) for the Chrome browser.



Figure 1. Prototype Navigation



Figure 2. Graphics display elements

### 5.3.1 Tools Used

In the three partial evaluation sessions of the functional requirements for each application, developers were asked openly about all tools, guidelines and standards used as the basis of building each project. A summary follows as we show in Table 8.

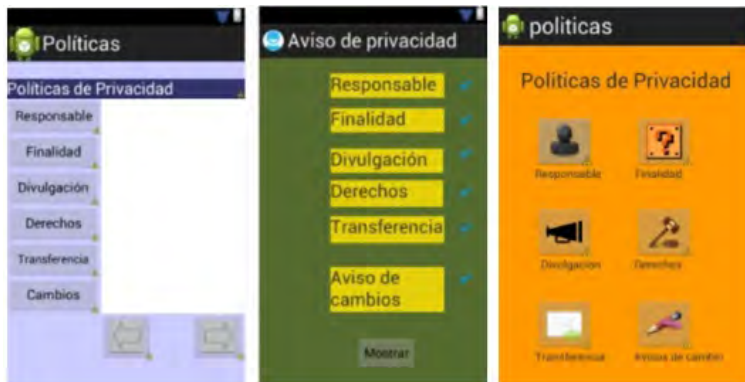


Figure 3. Displaying a Simplified Privacy Notice



Figure 4. Augmented Reality

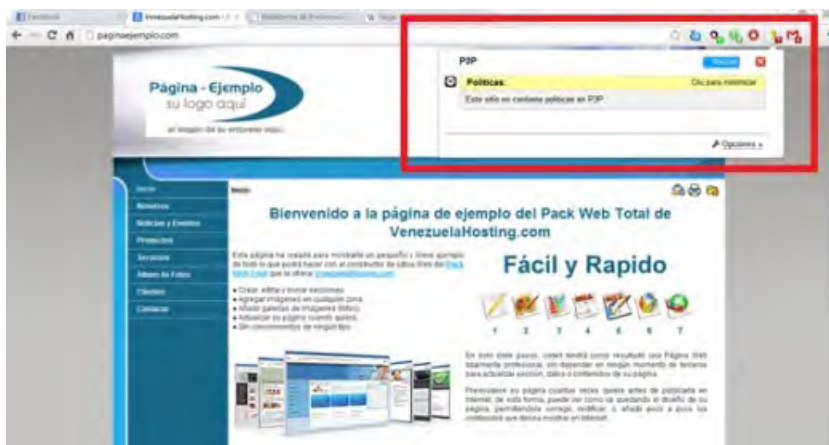


Figure 5. Display based on P3P

Table 8. Programming Tools, Guides And Standards

Job	Tools	Methodology	Information security standards
1	Eclipse	Modeling Scrum	Protection of personal data Law, collections of icons, password entry fields.
2	PHP and Eclipse	Modeling object-oriented	Summaries of Protection of personal data Law, Privacy Bird, data encryption function, password entry fields.
3	Eclipse	Object-oriented modeling	Protection of personal data Law, icons, password entry fields.
4	Java	Object-oriented modeling	Protection of personal data Law, APIs development.
5	PHP	Object-oriented modeling	Privacy-Bird-P3P * , XACML*** <a href="http://www.w3.org/P3P/">http://www.w3.org/P3P/</a> ** <a href="http://sunxacml.sourceforge.net/">http://sunxacml.sourceforge.net/</a>

5.3.2 Interaction Pattern: Privacy Policy

For this project, the following interaction pattern was designed based on the results in Section IV.2 and previously published work: [19, 14].

Name
Full privacy notice
Problem
The Privacy Notice is a document generated by the individual or entity responsible for the proper compiling and processing of personal data and should be made available to the owner of the data. This document does not have a defined structure and each party decides on the format. The document is written in legal language, unclear to the end user, and it is difficult to identify its parts. It is commonly ignored by users because of its length.
Solution
A hierarchy of data distribution and navigation information based on usability rules is proposed. The user is shown the procedure to identify the elements of a privacy notice and the function of each one.
Context
According to Article 16 of the FPDPPP (Federal Law on Protection of Personal Data Held by Private Parties) in Mexico, comprehensive format is required to explicitly include the following:
Handling of personal data collected by the responsible
Individuals responsible for gathering personal data
How to exercise ARCO rights by the owner of personal data
How changes in the Privacy Notice will be announced to personal data owner
How to limit the use of personal data
Consent for the transfer of personal data
Revoking consent for processing captured personal data
Revoking consent for the transfer of data to third parties
Explicit information about vulnerable groups

This interaction pattern continues on the following page————>

---

Use of Cookies and additional data collection technologies  
Display of consequences of possible misuse of data  
Tips to prevent misconduct of the personal data

Usability  
Helps users to identify the elements of a Privacy Notice.

Consequences  
Users can identify the parts of a Privacy Notice, recognize its importance and exercise their ARCO rights in a timely manner.

Related patterns  
Personal information  
Sensitive personal data  
Exercise ARCO rights

Example  
In Figure 6 we show an example.



Figure 6. Privacy Policy Example

---

## 5.4 Utility tools

### 5.4.1 AssessmentTools

At the last review session participants were asked to evaluate the tools they used based on the following Likert scale:

- 1 - Strongly Disagree
- 2 - Disagree

- 3 - Neither agree nor disagree
- 4 - Agree
- 5 - Strongly Agree

Research question: “Tools, guidelines and standards that you used in your project significantly facilitated the following tasks.” We exhibit results in Table 9.

**Table 9. Three tasks with likert results**

Job	Programming a privacy app	Designing privacy GUIs	Consulting privacy icon
1	3	2	3
2	3	2	3
3	4	3	4
4	3	3	3
5	2	2	3
Average	3	2.4	3.2
Utility tool for developers	3/5 = 60%	2.4/5 = 48%	3.2/5 = 64%

Participants were also asked to respond to the following “Question 1” and “Question 2” as a team:

Is there a universal language that represents computer security issues and end user privacy?

- Is the participation of graphic designers pertinent for building these interfaces?
- In Table 10 we show answers.

**Table 10. Additional elements**

Job	Question 1	Question 2
1	Isolated elements	Yes
2	No	Yes
3	No	Yes
4	No	Yes
5	No	Yes

### 5.4.2 Impact of Interaction Patterns

During the Summer term of 2013, students who developed Applications 1, 2, 3 and 4 (Described in Section V.C “Study Projects”) studied the concept of interaction patterns,



unknown to the students at that time. They were introduced COPPEMA[14] and Privacy Policy pattern (Seen in Section V.1) were presented. They were asked to estimate how the projects would have been influenced had they been given to them at the beginning of their projects, based on the following Likert scale:

- Strongly disagree
- Disagree
- Neither agree nor disagree
- Agree
- Strongly Agree

Study question: “Has a collection of Privacy interaction patterns significantly facilitated the tasks associated with the application development presented in the Fall term of 2012?”. In Table 11 we exhibit the results:

**Table 11. Tasks and associated rating**

Job	Programming a privacy app	Designing privacy GUIs	Consulting privacy icon
1	4	5	4
2	4	5	5
3	4	5	4
4	4	4	4
Average	4	4.75	4.5
Utility tool for developers	4/5 = 80%	4.75/5 = 95%	4.5/5 = 90%

Finally, they were asked to estimate the average hours they would have saved if they had been given the interaction patterns at the beginning of their projects. In Table 12 we show the answers.

In Table 13 we present a comparison between the use of legacy development tools they freely choose and interaction patterns follows.

### 5.4.3 E. Results of the Second Part

Neither team used tools to build prototypes, nor was any comprehensive collection of specialized items identified as having helped them in their design and development. Supporting elements (Java classes, extensions for particular types of browsers) were found to support any type of application, as seen in Table 8. In Table 9 we exhibit the tools used are considered, in general, 60% useful in designing a privacy application, compared to 80% for interaction patterns as we show in Table 11.

Table 12. Estimated impact of interaction patterns on application design

Job	Average real hours invested	Average estimated hours with interaction patterns
1	20	15
2	10	7
3	12	7
4	28	23
Total hours	70 hours	52 hours

Table 13. Interaction patterns versus legacy tools

	Programming a privacy app	Designing privacy GUIs	Consulting privacy icon
General development tools	60%	48%	64%
Interaction patterns	80%	95%	90%

In Table 10 the existence of a universal language in the area of security information is not recognized and the participation of a graphic designer is considered pertinent in these projects. In Table 12 we exhibit an average reduction of 25.7% of the total development time for applications is expected if interaction patterns are made available at the beginning of the project.

Traditional design tools are useful for creating IT security applications. However, having a collection of interaction patterns makes it easier for software developers, as we can show in Table 13.

## 6. Discussion

Future studies could include graduates of these majors that are involved in the software industry and graphic design students who have experience in software development teams. In Table 9 we exhibit developer satisfaction as regards the use of traditional security tools for graphic interfaces design is very low. This fact opens up opportunities to suggest models that support these projects. Due to the increasing number of Internet users, it is necessary to build more security software applications. Measuring the impact of applying cognitive ergonomics as a global framework in the design of interaction patterns is a broad line of research. In this work the specific end user interface elements (human factors) were measured according to how they improve the performance of the developer. Future work could measure end user satisfaction of the interaction pattern, specifically items such as mental workload compared to

mental fatigue, the time to complete the whole privacy notice recognition with and without the cognitive ergonomics elements.

In Table 8 we exhibit Legacy development tools support the building of effective applications; however, having a collection of interaction patterns facilitates the tasks involved in privacy design. In Table 12 and Table 13 we show these patterns are presented as good practices in the software development cycle and reduce the time spent. The fact that there is no universal language of security information opens up the playing field to new ideas that will enhance collaboration between experts and multidisciplinary teams as we can see in Table 10.

Building up a collection of interaction patterns that includes these results can influence issues of usability and user experience, to build tools to support privacy management. In this work the interaction pattern was designed with the visual thinking cognitive ergonomics but it was not explained to the developers. Many applications are developed only by security experts, and this means that they fulfill functional requirements, but are far from solving real problems for users.

## 7. Conclusion

Current interaction with the elements of a privacy policy does not facilitate the exercise of ARCO rights, making Internet users browse in insecure mode, at the expense of any possible misuse of personal information.

The goal of presenting users with data handling policies is met from a technical and legal framework, but the ultimate goal, which is to protect people and facilitate the exercise of ARCO rights, is not satisfied. Registered user experience as to privacy policies is not positive. Items that facilitate the design and construction of interfaces involving privacy issues for end users are scarce. To implement this kind of applications, developers can apply any methodology and software development tools, depending on needs and available resources; however, once they have a set of specific interaction patterns, the issues involved are reduced.

The study suggests that future work should propose collections of interaction patterns that include examples based on how people learn and interact with human-computer interfaces. This would improve the information security culture of Internet users, reduce risks and mitigate associated problems such as mental fatigue, boredom and exceeding mental workload limit.

## 8. References

- [1] L. Rebollo, «Vida privada y protección de datos. Un acercamiento a la regulación internacional europea y española,» [En línea]. Available: <http://biblio.juridicas.unam.mx/libros/6/2758/10.pdf>. [Último acceso: 2014].
- [2] L. Cranor, M. Langheinrich y M. Marchiori, «W3C The Platform for Privacy Preferences 1.0 (P3P1.0) Specification,» 2002. [En línea]. Available: <http://www.w3.org/TR/P3P/>. [Último acceso: 2013].
- [3] J. King, A. Lampine y A. Smolen, «Privacy: Is there an app for that? Symposium On Usable Privacy and Security,» [En línea]. Available: <http://dl.acm.org/citation.cfm?id=2078843>. [Último acceso: 2013].
- [4] S. Garfinkel y L. Faith, Security and Usability. Designing Secure Systems That People Can Use., O'Reilly , 2005.
- [5] P. Gage, L. Cesca, J. Bresee y L. Faith, «Standardizing Privacy Notices: An Online Study of the Nutrition Label Approach,» [En línea]. Available: [http://www.cylab.cmu.edu/files/pdfs/tech\\_reports/CMUCyLab09014.pdf](http://www.cylab.cmu.edu/files/pdfs/tech_reports/CMUCyLab09014.pdf).
- [6] H. Lindskog, Web Site Privacy with P3P, Wiley, 2003.
- [7] C. d. d. d. H. C. d. I. Unión, «Ley federal de protección de datos personales en posesión de los particulares,» 2010. [En línea]. Available: <http://www.diputados.gob.mx/LeyesBiblio/pdf/LFPDPPP.pdf>. [Último acceso: 2013].
- [8] J. Borchers, A pattern approach to interaction design, England: Wiley, 2001.
- [9] I. F. d. A. a. I. I. y. P. d. Datos, «Guía práctica para generar el aviso de privacidad,» [En línea]. Available: <http://inicio.ifai.org.mx/DocumentosdelInteres/privacidadguia.pdf>.
- [10] C. Fácil, «Generador de avisos de privacidad,» [En línea]. Available: <https://www.contratosfacil.com/document/51015f1c6df58/Generador-automtico-de-tu-Aviso-de-Privacidad-personalizado-en-minutos>.
- [11] P. Life, «HCI Patterns Collection - Version 2,» [En línea]. Available: [http://primelife.ercim.eu/images/stories/deliverables/d4.1.3-hci\\_pattern\\_collection\\_v2-public.pdf](http://primelife.ercim.eu/images/stories/deliverables/d4.1.3-hci_pattern_collection_v2-public.pdf). [Último acceso: 2013].
- [12] P. Gage Kelley, L. Cesca y L. Faith, «<http://repository.cmu.edu/cylab/1/>,» Standardizing Privacy Notices: An Online Study of the Nutrition Label Approach.
- [13] C. Jensen y C. Potts, «Privacy policies as decision-making tools,» SIGCHI conference on Human Factors in Computing Systems, pp. 471-478, 2004.
- [14] S. R. Murillo y J. A. Sánchez, «Studying the Relationships between the Management of Personal Data Privacy and User Interface,» Human Computer Interaction. Lecture Notes in Computer Science Volume 8278, pp. 79-89, 2013.
- [15] IEA, «International Ergonomics Association,» 1 January 2016. [En línea]. Available: <http://www.iea.cc/about/index.html>. [Último acceso: 20 June 2016].
- [16] D. Roam, La clave es la servilleta, Bogotá: Norma, 2009.

- [17] V. Ramachandran, *Phantoms in the brain*, New York: Harper Perennial, 1999.
- [18] N. K. Malhortra, *Investigación de Mercados*, Pearson Prentice Hall. 5ª. Edición., 2008.
- [19] S. R. Murillo y J. A. Sánchez, «Patrones de interacción y su aplicación a la privacidad en Internet,» de INCOSE, Puebla, 2013.

# Chapter # 8

## Analysis methodology for quality source code in software development

Carlos-Alberto Lóopez-López, Carolina-Rocío Sánchez-Pérez, Alberto Portilla y Marva-Angelica Mora-Lumbreras  
Universidad Autónoma de Tlaxcala (UATx)  
Facultad de Ciencias Básicas, Ingeniería y Tecnología  
Calzada Apizaquito S/N, Apizaco, Tlaxcala  
Emails: carlos.alberto.lopez@outlook.com, fkrlinasp, alberto.portilla, marva.morag@gmail.com

### 1. Introduction

Small and Medium Mexican Enterprises dedicated to software development have great competition in the market, the Ministry of Economy reports that Mexico currently has 755 certified centers under quality models such as the Capability Maturity Model Integration-CMMI, the NMX-059/01NYCE-2005 also known as Moprosoft, Team Software Process Performance and Capability Evaluation (PACE) TSPPACE [1]. Also in the framework of the sectorial agenda for the development of Information Technologies, federal government has defined as a strategy stimulating the IT market, linking the demand of economic sectors with the supply of products and quality IT services in Mexico [3]. This scenario implies that companies must ensure the quality of their products and processes to score a differentiator in the IT market to be highly competitive. To achieve this most adopted methodologies and models aimed at improving the quality of processes in the organization. However, this is not enough, as several of these models focus on obtaining documentation process which doesn't does not necessarily guarantee a quality product. The Moprosoft and CMMI models implement verification and validation practices for the processes documents of the area of Software Development and Maintenance. On the other side, Agile methodologies define practices such as peer review to the source code that focuses on issues such as format, indented, declaration of variables, etc., but not specifically define quantitative metrics to assess the quality of the source code. The quality of the source code is related to non-functional characteristics of the software, such as performance, maintainability, stability, security, to name a few, however many times to review the sour-

ce code we found that this presents some shortcomings such as repeated code, complex or not commented, which have an impact on non-functional characteristics even it meets the functionality requested by the customer.

The main purpose of software engineering is the fulfillment of user requirements and building quality software methodologically, the IEEE defines quality as the degree to which a system, component or process meets the requirements specified, or the degree to which a system, component or process meets the needs or expectations of the user [2]. Meanwhile, in quality models and process improvement methodologies emphasis on compliance with the process, and measurement it is done by adjusting the concept of quality process quality, yet the process of measuring source code quality focuses mostly on the contents of the documents produced by peer reviewers and not necessarily in a quantitative methodology where metric values must meet indicated values to be considered software quality.

This research focuses on the analysis of metrics to evaluate the quality of object-oriented source code and propose a set of thresholds to apply for quantitative indicators of the quality of the source code, this will allow to define a Methodology for source code analysis. The rest of the article is organized as follows, Section II presents related work, Section III introduces the software metrics, in Section IV we analyze software metrics object-oriented, Section V presents a comparison of thresholds, Section VI presents the Analysis Methodology and finally Section VII concludes the chapter and presents future work.

## 2. Related Work

In [4] metrics are proposed to measure the process, product and people (P3) in the generic stages of a development cycle: requirements, design and implementation. The intention is to highlight the value of measuring the complete development of a software product to ensure quality. For the requirements phase are explained metrics that focuses in on use cases to count the number of workers covered by a scenario requirements and quality metrics requirements that will allow measure no-ambiguity and that the cases are correct and complete. Examples of these metrics are a number of actors, messages and classes associated with the use case. For the design phase metric define both quantitative and qualitative, quantitative focus on the number of design elements, such as packages, classes, interfaces, methods, etc. Qualitative metrics to measure the quality of each design element such as: Instability, Abstraction, investment principle of dependency, dependency and acyclic Principle and Principle encapsulation principle. For the last phase of implementation developers coding requirements based on the design, metrics that are used at this stage to assess that the code is testable testable,

have the necessary comments, which is maintainable and portable. Examples of these metrics are the scope of methods, ramifications and symbols. While an important set of metrics defined in this chapter, it is concluded that the quality process must be specific to the organization providing a basis for product quality, research in this area is in continuous progress providing better metrics for the product, people and process, so as to support the development team.

On the other hand, there are different models and quality metrics that support the development of a software product, which allows the final product quality. In [5] the need for quality metrics from different views it is studied. It seeks to examine why software metrics are required and revisit their contribution to the quality and reliability of software. The authors propose a classification process metrics, project metrics and product metrics. Process Metrics are oriented to development process development, covering aspects such as the duration of the process, the cost involved and the type of methodology used. Meanwhile, project metrics are focused on monitoring the status of a project, so that, they can minimize the risks associated with this. Finally, product metrics focuses on the attributes of the software product in the development phase. The metrics considered in this work are: program size, complexity of software design, performance, portability, maintainability, coinciding with measuring aspects that define Poornima and Suma work. In [5] presents a comparison of the strengths and weaknesses of software metrics, based on the idea that the industry has no standards and measurement practices. The strengths and weaknesses of the physical and logical lines of code, function points, as well as object-oriented metrics are highlighted, this review will allow a discrimination of those considered for our analysis.

Finally, in the object-oriented paradigm, they are still using metrics that were created for a procedural approach paradigm. However, have emerged different sets of metrics to measure the characteristics of object-oriented systems, it is necessary to identify which of these sets of metrics are those that provide a better measure of product quality. In [6] intends to evaluate three different sets of metrics for object-oriented paradigm:

- Metrics for Object Oriented Design MOOD
- Shyam R. Chidamber and Chris F. Kemerer
- Lorenz and Kidd

As a result of the comparison of these three groups, metrics authors recommend the use of the first two group of metrics, which can measure the object-oriented features, such as encapsulation, inheritance and polymorphism.



### 3. Software Metrics

There is a set of software metrics for object-oriented paradigm, in this section the metrics used in this research are defined. These metrics are those that after an analysis seem better suited for the type of projects and methodologies that we use for software development projects.

#### 3.1 Cyclomatic complexity

Cyclomatic complexity measures the logical complexity of a program software. This metric gives great added value, since it is possible to use during software development when maintenance is necessary, to identify the tests and reengineering needed.

#### 3.2 Size code

This metric was one of the first to emerge, as a quick way to measure a program. Many metrics use it as the basis for their measurements. There are different measurements that will be made to the source code, below are some:

- Total number of lines.
- Number of lines of code (not counting line spaces and comments).
- Number of methods.
- Number of sentences.
- Number of lines of comments.

#### 3.3 Documented code

A good practice when coding a program is to add enough information to explain what the code does. To ensure that the code contains the necessary documentation must add comments to the class, methods, global variables, loops, conditions and code fragments where the intention is not evident.

#### 3.4 Weighted Methods per Class (WMC)

The WMC metric [11] is the sum of the complexities of all methods of a class. It is an indicator of the effort required to develop or maintain a particular class.

$$WMC = \sum_{i=1}^n Ci$$

Where a class  $C_i$  has methods  $M_1, \dots, M_n$ , with their respective complexity,  $C_1, \dots, C_n$ .

### 3.5 Response for Class (RFC)

The RFC metric [11] is the number of different methods and constructors invoked when a method of a class runs.

$$RFC = |RS|$$

Where RS is the set for response for class.

$$RS = \{M\} \cup \bigcup_i \{R_i\}$$

Where  $\{R_i\}$  is the set of methods called by method  $i$ ; and  $\{M\}$  is the set of all methods in class.

### 3.6 Lack of Cohesion of Methods (LCOM)

The LCOM metric [11] is responsible for measuring the lack of cohesion of a class. Cohesion measures the degree of specialization of a particular class. Classes with a high degree of cohesion are easier to maintain, and as their purpose is more specific and tends to be more reusable. Consider a class C1 with  $n$  methods  $M_1, M_2, \dots, M_n$ . Be  $\{I_j\}$  = the set of variables instances by the method  $M_i$ .

There are sets  $n$  such that  $\{I_1\}, \dots, \{I_n\}$ ; Be  $P = \{(I_i, I_j) \mid I_i \cap I_j = \emptyset\}$ , and  $Q = \{(I_i, I_j) \mid I_i \cap I_j \neq \emptyset\}$ . If all sets  $n$   $\{I_1\}, \dots, \{I_n\}$  are  $\emptyset$ , so  $P = \emptyset$ .

$$LCOM = \begin{cases} |P| - |Q| & \text{if } |P| > |Q| \\ 0 & \text{otherwise} \end{cases}$$

### 3.7 Coupling Between Objects (CBO)

The CBO metric [11] is responsible for measure is coupled few classes a class, a class is coupled to another when you call methods of a class or use its attributes.

### 3.8 Depth of Inheritance Tree (DIT)

The DIT metric [11] is aimed at identifying the level at which is a class in the inheritance hierarchy. A class that is at a very low level of the hierarchy is more difficult to identify its operation, having to know the function of the classes in on a higher level.

### 3.9 Number of Children (NOC)

The NOC metric [11] measures the number of classes are inheriting a specific class. This information is useful when you want to change a class, measures the impact of knowing the number of classes that inherit from it.

### 3.10 Duplicated code

When coding a program usually falls in a very bad practice, to copy blocks of code. When copying the code there are the following disadvantages:

Increases code size, a larger code size complicates maintenance.

- When changes are necessary, must change all the duplicate code, and if for some reason any section is not updated on which a copy was made, the portion of code can be inconsistent and cause the system to malfunction.
- The copy code reflects the lack or deficiency in the design of software.
- For these reasons it is advisable to avoid duplicated code in a software program.

Table 1. Metric's comparison

Metric	Works related			
	[7]	[4]	[5]	[6]
Cyclomatic complexity	YES		YES	
Size code	YES		YES	
Documented code	YES			
Weighted Methods per Class	YES	YES		YES
Response for Class	YES	YES		YES
Lack of Cohesion of Methods	YES	YES		YES
Coupling Between Objects	YES	YES		YES
Depth of Inheritance Tree	YES	YES		YES
Number of Children	YES	YES		YES
Symbol Coverage Metric		YES		
Method Coverage Metric		YES		
Branch Coverage Metrics		YES		
Method Hiding Factor				YES
Attribute Hiding Factor				YES
Metric Inheritance Factor				YES
Attribute Inheritance Factor				YES
Polymorphism Factor				YES

This table continues on the following page————>

Metric	Works related			
	[7]	[4]	[5]	[6]
Coupling Factor				YES
Class Size Metrics				YES
Class Inheritance Metrics				YES
Class Internals Metrics				YES

## 4. Metrics Analysis

In this section present a comparative analysis of the metrics defined in the previous section. The Table 1 shows a comparison of selected metrics associated with the work where they were implemented. The first nine metrics are the most mentioned and implemented, which is an indicative of the importance to measure the quality of object-oriented source code. To get a broader picture of the advantages and disadvantages of these metrics, for each metric an analysis is performed to define the values or thresholds recommended in this work.

### 4.1 Cyclomatic complexity

When measuring a program an important step is to have reference parameters to identify the portions of code where the reduction of complexity must be made. Table 2 shows the ranges that a program must meet, based on its complexity, this will serve as a basis for measuring the quality of the source code.

### 4.2 Weighted Methods per Class

A class with a high WMC indicates that the class is complex, so it is difficult to reuse and maintain. This metric uses the cyclomatic complexity to calculate the complexity of each method. Rosenberg [9] defines that the upper limit to this metric is 100, lower values will be acceptable. Based on this threshold, each method has a complexity maximum of 10; that is a simple program and applying the rule of thirty in code size, the following thresholds are proposed in Table 3.

Table 2. Ranges program complexity [8]

Cyclomatic Complexity	Risk Evaluation
1-10	a simple program, without much risk
11-20	more complex, moderate risk
21-50	complex, high risk program
mas de 50	untestable program (very high risk)

**Table 3. Weighted methods per class ranges**

Weighted Methods per Class	Method	Evaluation
1-100	1-10	Desirable
101-200	11-20	Acceptable
201-300	21-30	Not recommended

**Table 4. Response for class ranges**

Response for Class	Evaluation
1-50	Desirable
51-100	Acceptable
more than 100	Not recommended

**Table 5. Coupling between objects ranges**

Coupling Between Objects	Evaluation
0-5	Desirable
6-7	Acceptable
mas de 7	Not recommended

### 4.3 Response for Class

RFC value of a class should not exceed 50, although it is acceptable to have values of 100 as a limit. The Table IV shows these ranges.

### 4.4 Lack of Cohesion of Methods

This metric has a well-known threshold, LCOM4 must be equal to one in a well-designed class. When a class has values greater than one, it has to be refactored to have more specific tasks.

### 4.5 Coupling between Objects

Is the number of classes to which a given class is coupled? Rosenberg [9] defines a threshold of 5. Taking this threshold as reference to the thresholds presented in Table 5.

Table 5. Initial values metrics

Class	Linesofcode	Method	ComentLines	WeightedMethodsperClass	ResponseforClass	Duplicatedcode
RegistroUsuario	295	11	1	55	49	67
CambiarContrasenía	94	3	0	10	21	33
CancelarCuenta	84	3	0	10	20	33
Authenticator	128	3	3	7	19	0
GestorArchivosBussines	221	9	0	34	52	43
InicioSesionBussines	182	5	16	64	132	0
RegistroUsuarioBussines	559	21	10	71	108	74
CambiarContraseníaBussines	51	2	0	8	12	16
CancelarCuentaBussines	42	2	0	6	11	16
Utilidades	193	14	8	65	57	0
Total	1849	73	38	330	481	282

## 4.6 Size code

This metric is easy to obtain and gives an initial overview of the current state of a software program. There are several controversies regarding the importance of code size, some authors propose to focus on other metrics and others recommend paying special attention.

Stefan and Martin [10] proposed the Rule of 30, which states:

- Methods should not have more than an average of 30code lines (not counting line spaces and comments).
- A class should contain an average of less than 30methods, resulting in up to 900 lines of code.

- c. A package should not contain more than 30 classes, comprising up to 27,000 code lines.
- d. Subsystems with more than 30 packages should be avoided. Such a subsystem would count up to 900 classes with up to 810,000 lines of code.
- e. A system with 30 subsystems would thus possess 27,000 classes and 24.3 million code lines.
- f. If the system is divided into 3 to 10 layers, each layer comprises 3 to 10 subsystems.
- g. This rule serves as a benchmark, because there will be cases where it can not be applied.

## 4.7 Documented code

With regard to the thresholds for the amount of code documentation there are few recommendations; in order to have a benchmark we propose to take the number of methods that contains the class to define the number of comments lines, the comments of a method comprised of two-three lines. The following formulas are presented to define thresholds for code documentation.

$$LCCI = (M + 1) * 2$$

$$LCCS = (M + 1) * 3$$

Where LCCI is the lower limit of lines of code comments and M the number of methods of classes plus class comment. LCCS is the upper limit of lines of code comments.

## 5. Thresholds Assessment in a Study Case

Having identified the key metrics to measure and their thresholds, we proceeded to evaluate this values in a case study. In order to compare the thresholds of the metrics discussed in the previous section the measurement of a software system was made, the system used allows registration of requests for procedures. The SonarQube tool was used for measurement.

The system case study is composed of 157 classes in the Java language, divided into 10 modules. The Tables 6 and 7 show baseline measurement of ten classes that make up the registration module, specific user account with SonarQube. The effort invested in measuring and refactoring of this module was 40 hours of a developer, this time dedicated to improving the source code will be a benefit as necessary to maintain the module.

**Table 6. Unique values metrics**

Metric	Value
Lack of Cohesion of Methods	1
Number of Children	0
Depth of Inheritance Tree	1

As we can see, the first metric to be improved is the documentation of the code, since only the 30% of source code is documented, when the recommendation is to have 100% of classes commented.

With regard to the cyclomatic complexity all classes are within the desirable threshold limit, that is 100. The metric for a class response has acceptable values except for a class that throws a very high value and is recommended to decrease this value to less than the 100 desired range.

This module has a 15% duplicated code, which is recommended to decrease or eliminate.

The Table VII shows metrics that yielded the same value in all classes. The lack of cohesion metric methods is complying with the rule that must be equal to one and the other two metrics show very low values, which is an indication that they are not taking advantage of the properties of inheritance.

Once the measurement and identification of possible improvements, we proceeded to the modification of the source code to improve quality and add new features. The `RegistroUsuarioBussines.java` class was refactored resulting in the `MovimientoFamiliaBusiness.java` and `EnvioCorreoBusiness.java` classes to get a better specialization in classes, thereby reducing the cyclomatic complexity in these classes. In order to reduce duplicate code, was found that the class `CancelarCuentaBussines.java` and `CambiarContraseniaBussines.java` were unified classes and were doing similar tasks, which resulted in the `ModificacionUsuarioBusiness.java` class that implements the similar class.



Table 8. Improved metric values

Class	Linesofcode	Method	ComentLines	WeightedMethodperClass	ResponseforClass	Duplicatedcode
RegistroUsuario	317	16	37	59	51	0
CambiarContrasenia	116	5	7	11	26	39
CancelarCuenta	99	4	6	10	24	39
Authenticator	155	5	3	6	15	0
GestorArchivosBussines	269	9	0	34	49	51
InicioSesionBussines	224	5	34	17	128	33
RegistroUsuarioBussines	336	9	30	40	61	27
MovimientoFamiliaBusiness	232	6	23	22	52	133
EnvioCorreoBusiness	212	6	23	18	35	0
ModificacionUsuarioBusiness	182	5	17	18	31	28
Utilidades	138	13	41	33	45	0
Total	2280	83	221	268	517	350

Values with improvements to the code and analyzing metrics are shown in Table 8, it should be noted that some metrics increased by new features that were added to the system.

As you can see code documentation improved is close to 90% and can raise more, still there is a missing class to document.

Another metric that improved is the cyclomatic complexity when performing refactoring process and there is the possibility of reducing their values in a class.

Response for a class and duplicated code metrics did not improve, and its increase was proportional to the code that was added by new functionalities. It is advisable to do further analysis on this code, maybe adding generic methods to handle classes and avoid duplicate code.

The Metrics defined in Table VII maintained the same values in all classes analyzed module. It can be seen that by making the specialization and refactorizacion refactoring

sections based on thresholds analyzed served to enhance their values in most classes code.

## **6. Analysis Methodology**

Measuring the source along with the comparison of thresholds give an overview of the usefulness of having metrics and thresholds, which are the basis for the definition of the measurement methodology source. In this section, we give the steps to follow in the methodology to guarantee the source code quality.

### **6.1 A. Define the scope of measurement**

The first step is applying this methodology to a software project, there must be established if the entire system will be evaluated or only apply to a part of this. To make this decision we have to take into account the time and the number of developers required to improve the source code.

### **6.2 Measure the source code**

Once defined the scope of improvement proceeds to measurements to obtain source code metrics values set by the methodology.

- Size code
- Documented code
- Weighted Methods per Class
- Response for Class
- Lack of Cohesion of Methods
- Coupling Between Objects
- Depth of Inheritance Tree
- Number of Children
- Duplicated code

### **6.3 Threshold comparison and identification of improvements**

Values obtained against each metric thresholds are compared. Classes that have values that are not patched ranges will be candidates to improve according to the metric that do not comply.

## 6.4 Source code Improvement

*Size code:* When the limit is exceeded the only solution is to refactor the class along with the identification of duplicate code unifying in a new class.

*Cyclomatic complexity:* To reduce the complexity of a method, one can make use of utilities that perform tasks that are repetitive and separating tasks in new methods.

*Documented code:* For this metric the solution is obvious we must add the necessary comments presented in Section 4.

*Duplicated code:* You have to identify duplicate code blocks and make a unification on these blocks in a single method.

There will be cases where the project cannot meet the thresholds and where one must justify why the thresholds are not in recommended ranges. Given that the thresholds are not law; they are only a framework.

## 7. Implementation

### 7.1 Case study

The Institutional Information Management System (Sistema Institucional de Informacion Administrativa SIIA) of the Autonomous University of Tlaxcala encompasses a large number of processes at the same university, for example, Student Services, Human Resources, Financial Resources, Heritage, Libraries, Tutoring, to name a few.

SIIA is composed of 2,211 java classes contained in packages 280, with a total of 323,126 lines of code.

### 7.2 Implementation methodology plan

It is intended to implement the methodology to the entire SIIA, in order to have a source code of quality and maintainable, once achieved this aim migrate this system to a newer platform with modern technologies which are responsive to mobile devices such as smartphones and tables tablets.

### 7.3 Implementation of the methodology

Define the scope of measurement: The implementation will gradually start with School Services module, which is composed of the following sub-modules:

- Student Administration
- Change of status of students
- Ratings
- Course record
- Credentialing
- Official documents
- Registrations of students
- Educative offer
- Plans and programs of study

Measure the source code: In Table 9 and 10 the measurement values are presented.

Threshold comparison and identification of improvements: Values that exceed the thresholds of the metrics are shown in Table 11.

Table 9. Metric values

Class	Lines of code	Methods	Comment lines	Weighted Methods per Class	Response for class	Duplicated code
AdminBajasActionBean	152	11	0	26	74	0
AdminBajasEJBBean	364	20	1	50	103	42
CapCalificacionesBean	546	10	6	58	142	406
AutorizarCargaBean	238	11	0	38	52	66
AutorizarCargaEJBBean	284	11	0	23	32	0
RealizarCargaBean	782	14	0	134	104	117
AdminSolCredBean	390	12	35	36	99	17
ImpCredActionBean	691	19	94	79	108	34
ValCredActionBean	152	9	0	26	43	38

This table continues on the following page————>

Class	Lines of code	Methods	Comment lines	Weighted Methods per Class	Response for class	Duplicated code
CargaMasCredBean	176	10	0	24	51	52
AdminSolCredEJBBean	1674	39	235	143	71	106
ValCredEJBBean	262	12	5	38	60	17
CargaMasCredEJBBean	194	8	16	36	44	20
ImpDocOfiBean	2215	27	7	366	236	395
ConstanciaEstudiosPdf	147	2	1	11	46	91
ConstanciaExpComplPdf	149	2	4	11	50	92
GenDocOfiActionBean	54	4	0	4	13	0
GenDocOfiEJBBean	268	10	8	46	35	68
AdminOfertaBean	1223	50	5	197	237	109
AdminOfertaEJBBean	1784	59	26	187	110	532
Total	11745	340	443	1533	1710	2202

**Table 10. Unique values metrics**

Metric	Value
Lack of Cohesion of Methods	1
Number of Children	0
Depth of Inheritance Tree	1

**Table 11. Values that exceed the thresholds**

Class	Lines of code	Methods	Comment lines	Weighted Methods per Class	Response for class	Duplicated code
AdminBajasActionBean			0			
AdminBajasEJBBean			1	103		42

This table continues on the following page →

Class	Lines of code	Methods	Comment lines	Weighted Methods per Class	Response for class	Duplicated code
CapCalificacionesBean			6		142	406
AutorizarCargaBean			0			66
AutorizarCargaEJBBean			0			
RealizarCargaBean			0		104	117
AdminSolCredBean						
ImpCredActionBean					108	
ValCredActionBean			0			38
CargaMasCredBean			0			52
AdminSolCredEJBBean	1674	39				
ValCredEJBBean			5			
CargaMasCredEJBBean			16			
ImpDocOfiBean	2215		7	366	236	395
ConstanciaEstudiosPdf			1			91
ConstanciaExpComplPdf			4			92
GenDocOfiActionBean			0			
GenDocOfiEJBBean			8			
AdminOfertaBean	1223	50	5		237	
AdminOfertaEJBBean	1784	59	26		110	532

## 8. Conclusions and Future Work

One of the elements to assess various methodologies that ensure quality software development is the analysis of code. The idea of proposing an objective assessment of the source code, through quantifiable metrics is to help that the evaluation process under any methodology (CMMi, MOPROSOFT, etc) is carried out efficiently. By having well-defined processes for evaluating the code effort one organization can focus on other areas less feasible to be processed automatically processes such as planning processes, monitoring and control, requirements management, among others.

When it is intended to ensure the quality of the source code, having a set of metrics and their respective reference thresholds can help to start with the modification or correction of the source code. By having quantitative values one can identify the system's

source code that has a red light that could affect future aspects of maintainability, performance, efficiency, or others quality requirements of the system. It is noteworthy that these thresholds are not law and there will be cases where it can not be within acceptable ranges, in this cases would have to reach a compromise on the values defined.

To give continuity to this research work is to strengthen the methodology perfecting phase of improving the source code. Also, to run the implementation of this methodology in a software system that uses the heritage property in order to define the thresholds of the metrics that measure this property.

## 9. References

- [1] Padron de Empresas con niveles de Calidad, México. Secretaría de Economía, enero, 2016. VALUES THAT EXCEED THE THRESHOLDS
- [2] R. S. Pressman Ingeniería del Software, Un enfoque práctico, 7a ed. Mc Graw Hill. 2010.
- [3] Agenda sectorial Prosoft 3.0, México. Secretaría de Economía, Enero, 2016.
- [4] U. S. Poornima and V. Suma, Significance of Quality Metrics during Software Development Process, International Conference on Innovative Computing and Information Processing (ICCIP - 2012), 2012.
- [5] M. S. Rawat, A. Mittal and S. K. Dubey, Survey on Impact of Software Metrics on Software Quality, (IJACSA) International Journal of Advanced Computer Science and Applications, 2012.
- [6] A. K. Sharma, A. Kalia and H. Singh, Metrics Identification for Measuring Object Oriented Software Quality. (IJSCE) International Journal of Soft Computing and Engineering, 2012.
- [7] L. H. Rosenberg and L. E. Hyatt, Software Quality Metrics for Object Oriented System Environments. Crosstalk Journal, Software Technology Support Center, 1997.
- [8] M. Bray, L. Martin, C4 Software Technology Reference Guide, Software Engineering Institute, 1997.
- [9] L. Rosenberg, R. Stapko, A. Gallo, Object-Oriented Metrics for Reliability, Presentation at IEEE International Symposium on Software Metrics, 1999.
- [10] M. Lippert, S. Roock, Refactoring in Large Software Projects: Performing Complex Restructurings Successfully, 2006.
- [11] S. R. Chidamber, C. F. Kemerer, A metric suite for object oriented design, IEEE Transactions on Software Engineering, 1994.
- [12] C. A. Lopez, C. R. Sánchez, A. Portilla, M. A. Mora, Usando métricas para el análisis de calidad del código fuente, 4to. Congreso Internacional de Investigación e Innovación en Ingeniería de Software 2016, CONISOFT16, 2016.

# Chapter # 9

## The importance of functional and data requirements in supporting the adoption process of EHRS

Víctor H. Castillo, Leonel Soriano-Equigua, José L. Álvarez-Flores, Martha E. Evangelista-Salazar  
Facultad de Ingeniería Mecánica y Eléctrica  
Universidad de Colima  
Colima, México  
{victorc, lsoriano, alvarez\_jose, maevar}@uclm.mx

Ana I. Martínez-García  
Departamento de Ciencias de la Computación  
CICESE  
Ensenada, México  
martinea@cicese.mx

### 1. Introduction

Electronic health record systems (EHRS) are an important technology for providing quality and low-cost health care services [1]. Several efforts exist for describing open and standard EHRS architectures [2, 3], Figure 1 shows the OpenMRS architecture, which allows the implementation of EHRS. Nevertheless, EHRS has a limited adoption level, which negatively affects the provision of health care services [1]. Consequently, attempts for supporting the EHRS adoption process are important. Mainly, the adoption problem of EHRS it has been analyzed from a user perspective, e.g. physicians, nurses, and administrative staff. However, because of software engineers are significant stakeholders in the development process of EHRS, the adoption problem of EHRS also has been analyzed from a software engineer perspective [4]. Beginning with this perspective it is possible to analyze the software artifacts for promoting the adoption of EHRS these systems. From the stages of software engineering process, the requirements engineering stage is critical for helping to understand and define required services from a system. Proposals for supporting adoption of EHRS with information systems (IS) are scarce. In [5] is described the use of IS for this kind of support, however, in this work it is only depicted design considerations, but it does not describe the software engineering viewpoint. Accordingly, a software engineering perspective about



requirements of automatic support for assisting the adoption of EHRs would reveal a unified point of view about software models for facilitating a greater likelihood of EHRs adoption.

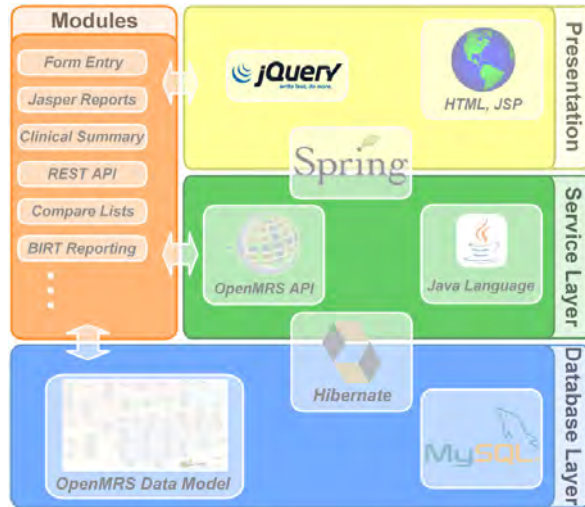


Figure 1. The OpenMRS architecture [2].

The objective of this study is to prioritize the IS' requirements that the engineers estimate as necessary for supporting the adoption process of EHRs. The article is organized as follows. The second section describes the central role of the requirements in the provision of support to adoption of EHRs. Then, in the third section, we depict the methodological approach for carrying out the prioritization of requirements to automatically support the adoption of EHRs by physicians. Finally, in section four we describe a conclusion about our work.

## 2. Previous Work

From an interaction design perspective [6], requirements can be classified into four important types: 1) functional, for denoting what the software should do, e.g. a functional requirement for an EHRs that it should be able to manage medical records; 2) data, for specifying features of the required data, e.g. taking into account that the system under consideration is an EHRs, then the data must be accurate and safe; 3) context of use, which refers to the conditions (physical, social, organizational and technical environment) in which the interactive product will be projected to work; and 4) usability goals, which includes effectiveness, utility, learnability, and memorability.

Being an important aspect of the software lifecycle, requirements have also been studied from the perspective of EHRS. In [7] it is stated that some clinical data element requirements differed between health care settings, however, the study just explores a clinicians' perception. Hernández-Ávila et al. [8] identified experiences and best practices as a guide for gathering the requirements of EHRS, however, these elements were depicted from the analysis of perceptions from state health services officials and IT experts. Based on the American Medical Informatics Association (AMIA) for clinical decision support system roadmap [9], Kawamoto and Lobach [10] proposed a service oriented architecture for EHRS fulfill business requirements, this architecture is founded on diverse EHRS stakeholder's viewpoint, which does not include software engineering position.

Kuperman et al. [11] enact content specifications for a Nationwide Health Information Network Trial Implementations. This work defines specifications for implementing EHRS, but it is focused on the process for developing the content specifications from a user standpoint. Several works study EHRS interaction design approaches, but also from the user perspective [12, 13]. Another work [14] depicts EHRS' design specifications, this considers the viewpoint from ambulatory EHRS vendors, clinical laboratories, physician organizations, government agencies, and the HL7 group. But this work does not describe the use of design specifications from a software engineer's view. Thereby, we can recognize in analyzed previous work a lack of studying and supporting the adoption process of EHRS from a software designer perspective. As we previously describe, this perspective is an important problem for providing health care quality services.

### **3. Method Description**

#### **3.1 Theoretical foundation**

In [15] is proposed a knowledge-based architecture for supporting the adoption of EHRS. Based on the relationship among critical factors for adopting EHRS and knowledge management processes, this architecture supports the development of systems for assisting the adoption of EHRS. This work enacts four system quality attributes for supporting the adoption of EHRS by physicians: communication among users, workflow impact, technical support, and expert support. The proposal of [15] is consistent with [5], in which critical factors for adopting EHRS would be an indicator aspect in the provision of automatic support for assisting the adoption of EHRS. In [5], the critical adoption factors for adopting EHRS by physicians are six: user attitude towards information systems, workflow impact, interoperability, technical support, communication among users, and expert support.

From previously explained, and considering that knowledge about an innovation is fundamental for supporting an adoption decision [16], we define a conceptual framework for supporting the adoption of EHRS by physicians, this is founded in two aspects: 1) the relationship among critical adoption factors of EHRS and information technology (IT) in knowledge management processes for assisting EHRS' adoption [5, 15]; and 2) the mapping of the adoption of innovations to a process of change [16, 17]. Our conceptual framework comprises four cognitive processes to automatically support the adoption process of EHRS by physicians. The processes are named "cognitive" because are based on necessary knowledge for supporting the adoption of EHRS by physicians.

### **3.1.1 "Contextualize" cognitive process**

First, the contextualize cognitive process helps the user to gather first knowledge about an innovation in the EHRS, which makes the user aware of this innovation [16, 17]. To be aware of an innovation is the first step for adopting it.

### **3.1.2 "Establish social network" cognitive process**

Secondly, the establish social network cognitive process presents user the necessary contacts for helping her/him to evaluate knowledge about the innovation (given to the user by contextualize process). Socializing an innovation is an important aspect of the adoption process of innovations [16, 17].

### **3.1.3 "Promote self learning" cognitive process**

On the other hand, the promote self-learning cognitive process provides the user with useful documents for decreasing uncertainty about an innovation. Self-learning also is an important aspect for promoting adoption of innovations [16, 17].

### **3.1.4 "Analyze user interaction" cognitive process**

At last, the analyze user interaction cognitive process evaluates if the user (a physician) was assisted by the other processes in the adoption decision of an innovation in the EHRS. This analysis helps to determine if the user adopts, or rejects, an innovation in the EHRS, e.g. a change in the EHRS' GUI. These cognitive processes helped for designing the experiment to evaluate our proposal.

## **3.2 Expert judgment based assessment**

Expert judgment is a formal approach to achieve information to specific questions about specific problems [18], e.g. defining requirements for automatic assistance to the adop-

tion of EHRS. The expert judgment approach has been used to assess and support conceptual frameworks in software engineering [19].

The aim of our conceptual framework is to lead the software engineers to define similar requirements for developing IS to support the adoption of EHRS. The consequence of this, after using the conceptual framework them would rank these requirements.

We used a scenario-based approach [20] to identify requirements for automatically support the adoption of an innovation in EHRS. As a result, we defined interaction scenarios in which we described how the cognitive processes would support the adoption process of EHRS. Also, each scenario depicts the way in which every cognitive process relates a critical adoption factor of EHRS to technology for support this critical factor. Figure 2 shows an interaction scenario in which a physician logs in the EMRS and observes that it has changed on the main screen. From these scenarios the software engineers would identify system requirements for automating each cognitive process.

Dr. Gómez is a physician who habitually uses the electronic medical record system. Today he accesses this system from his consulting room. When Dr. Gómez starts using the system at the beginning of his duties, he realizes that the system's main window has been modified and although the overall aspect of the system looks similar, he is a little confused. When Dr. Gómez explores the interface of the electronic medical record system, a warning from the system alerts him that the **Adoption assistant** of the electronic medical record system has a system tray notification for him. The physician looks into the notification and a window from the **Adoption assistant** opens up. When Dr. Gómez checks the window, he reads a notification that informs him that there has been a change in the user interface of the medical record, but that this change does not affect the structure of the electronic medical record. This knowledge makes him feel more secure about its use. In addition to this, the adoption support assistant enables the physician to access the electronic document that contains a user's guide to the electronic medical record system. So he opens up the guide and browses it, he screening the document and he realizes that the guide is much more detailed than the guide they had physically received when taking their training course and that it is also very updated, containing the explanation to the changes on the user interface he has just seen in the system. Likewise, after browsed the user's guide, Dr. Gómez realizes that the adoption support assistant provides him a list of physicians that are also users of the electronic medical record system and have some experience with the system, he is also provided with a list of technical support personnel and their contact information.

After reading the messages from the **Adoption assistant**, the first patient of the day arrives to the consulting room and Dr. Gómez treats him, glancing at his electronic medical record on the screen. He then captures the data from this new visit on the electronic medical record, which makes the accomplishment of his duties easier.

Figure 2. One of the interaction scenarios used for assessing the proposal.

### 3.2.1 Research instrument

We designed an instrument based on the elements of the cognitive processes. The instrument has 13 questions about system requirements, three about requirements for implementing each cognitive process and a final question about which would be the necessary technology for dissuading a reject decision of an innovation in the EHRS. The instrument is organized in the five sections depicted in Figure 3. Section A asks the participant for personal information, also, in this section the participant must link critical adoption factors of EHRS with IT in the knowledge management processes for supporting these critical factors (see Figure 4). Each participant was provided with a list of critical factors and IT to solve them. It also was provided with a list of interaction scenarios. Sections B, C, D and E of the instrument are organized as follows. In Part 1 (see subsections B.1, C.1, D.1 and E.1 in Figure 3) the participant is asked to relate the events described in the scenarios (e.g. a physician is aware of the GUI modification and he needs expert support) with IT that can support each cognitive process (e.g. knowledge directories for supporting social interaction), as it is showed in Figure 4. Then, in Part 2 (see subsections B.2, C.2, D.2 and E.2 in Figure 3) the participant must specify the type of knowledge representation structure (e.g. a programming flag specifying if a physician has acceded an electronic document for contacting him with experts) that could support the implementation of each relationship specified in Part 1 (see Figure 5). Next, in Part 3 (see subsections B.3, C.3, D.3 and E.3 in Figure 3) the participant must enact the system requirements for fulfilling each cognitive process. Additionally, when the participant is responding section E of the instrument (regarding to Analyze user interaction cognitive process), the participant must describe the helpful IT for dissuading a possible rejection decision of an innovation in the EHRS (Figure 6). The instrument was revised for the format of the questions and understandability of the wording. Also, this was reviewed by 2 independent researchers who had significant experience in the area of software development and adoption of EHRS. The instrument is available by requesting to the first author.

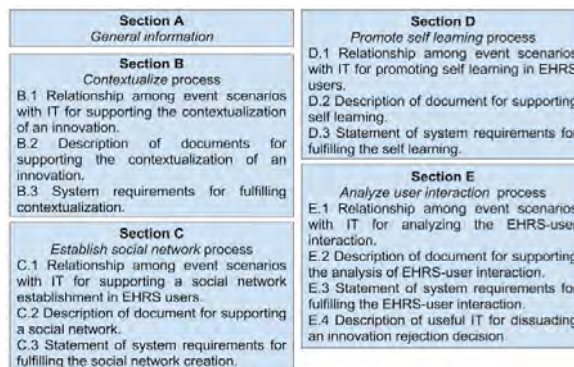


Figure 3. Instrument structure.

Tabla 1. Asociación de eventos de escenarios de problema con los roles de la administración de conocimiento

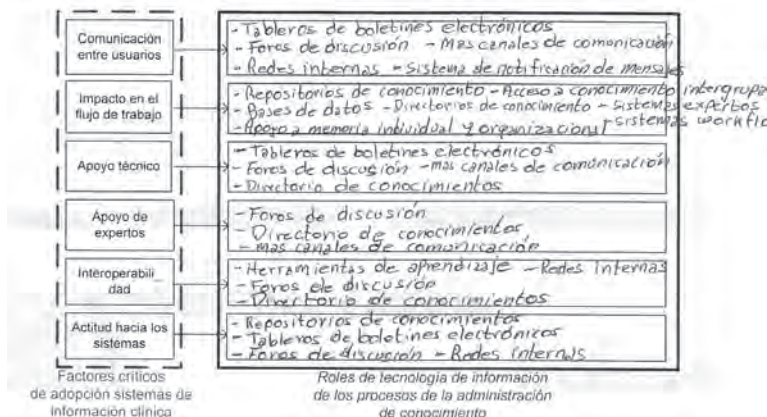


Figure 4. A section of the instrument. Relationship among critical adoption factors of EHRs and the information technology in the knowledge management processes

FORMATO II.3

Tabla 2. Estructura de representación información que ayude a promover el autoaprendizaje

Identificador	Atributo	Propósito
Registro de cada uno de los documentos leídos.	Falso - Verdadero	Una estructura que nos indica si el medico ha tenido acceso a los documentos necesarios para dominar el cambio.

Figure 5. A section of the instrument. A data structure for promoting self learning.

FORMATO II.4

Tabla 3. Tecnología de información que puede ser útil para disuadir decisiones de rechazo hacia TI<sup>1</sup>

No.	Rol de TI en los procesos de la AC <sup>2</sup>
1	<ul style="list-style-type: none"> <li>• Sistema de notificación de mensajes</li> <li>• Apoyo a memoria individual</li> </ul>
2	<ul style="list-style-type: none"> <li>• Repositorio de conocimientos</li> <li>• Directorio de conocimiento</li> <li>• Redes internas</li> </ul>
3	<ul style="list-style-type: none"> <li>• Combinación de nuevos recursos de conocimiento</li> <li>• Bases de datos</li> </ul>

Figure 6. A section of the instrument. Describing information technology for dissuading a rejection decision of EHRs innovation.

### 3.2.2 Expert selection

We based the expert selection process in [19]. The selection of experts for our study was based on the following four criteria: at least 1 year of software development; 2) wi-

llingness to act as impartial evaluator; 3) availability to commit needed time and effort; and 4) willingness to provide effective analysis, and interpretations. We used two modes of reaching potential respondents: personalized contact and professional referrals. We sent invitation emails to a selected pool of potential respondents.

### 3.2.3 Instrument administration

The assessment involved 10 participants from software development industry and educational institutions. All ten participants had at least 1 year of experience in software development. Four out of ten participants previously developed software in the medical domain, and eight out of ten had experience in all the fundamental software processes, i.e. software requirement specification, analysis, design, development and testing. The average experience in development tasks of participants was 6.7 years. All the participants attended a two hours assessment session.

Results analyzed in this chapter proceed from information collected from sections B.3, C.3, D.3 and E.3 in the instrument (Figure 3). To collect this information, we ask participants to answer an open question for each cognitive process. For example, to gather information about the cognitive process Promote self-learning (section D.3 in the instrument), we asked the following question “If you had to develop a system to promote self-learning among users of an electronic medical record, what requirements should fulfill that system?” (see Figure 7). Then, we coded their answers and defined categories, which are analyzed in next section.

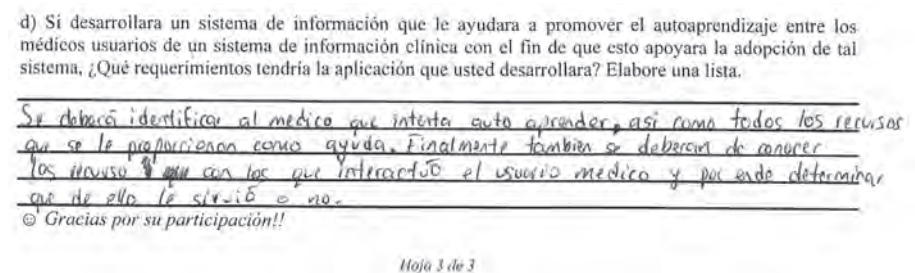


Figure 7. Instrument section for gathering information about the Promoting self learning the cognitive process.

## 3.3 Findings and discussion

The findings are founded on the frequencies of categories expressed by participants for each of the requirements arisen from the analysis of the interaction scenarios. Table I



describes the requirements most frequently expressed by software developers (column Most mentioned requirement), and the requirements less frequently expressed by them (column Less mentioned requirement).

**Table 1. Prioritized requirements by participants**

Cognitive process	Most mentioned requirement	Less mentioned requirement
Contextualize	Functional (9/10); data (7/10)	Usability goals (1/10); context of use (2/10)
Establish social network	Functional (10/10)	Usability goals (4/10); context of use (4/10); data (2/10)
Promote self learning	Functional (10/10)	Usability goals (1/10)
Analyze user interaction	Functional (10/10); data (7/10)	Usability goals (5/10); context of use (2/10)

The data in Table 1 were obtained as follows. First, we classified the open text-based answer of participants with regard to one of the four types of requirements we describe in section II (functional, data, context of use, and usability goals). Each participant generated four requirement lists similar to Figure 7, one list for each cognitive process. In total, we classified 40 lists, 10 for each cognitive process, and we count the occurrence of the type of requirement expressed by each participant. E.g. considering requirements specified by one of the participants (Figure 7), we incremented in 1 the counter of functional requirements, because of the participant describes 3 times the functional requirements: 1) to identify the physician who developed self learning; 2) to identify provided documents for developing self learning; and 3) to identify documents (resources) which a physician used for self learning. Some cases, one participant reported more than one kind of requirement. So, we counted requirements described by participants in each cognitive process, and we defined as a priority requirement the most common one. In Table I, some cases we cited two or three requirements as the most or less frequent, this means that its frequency is high or low with regard the other requirements. E.g. in the row Contextualize, the most cited requirements are the functional (9/10=90%) and data (7/10=70%); by the other hand, the less cited requirements are usability goals (1/10=10%) and context of use (2/10=20%). Another appointment is as follows. In the row Promote self learning (Table I), all the participants mentioned functional requirements (10/10=100%) as important for supporting this cognitive process, and the less mentioned requirement was usability goals (10% of participants described this kind of requirement). In the same row, we observe that no one described the context of use or data requirements as important for implementing the cognitive process Promote self-learning, and for this reason they are not included in the row.



Table 1 shows that functional, firstly, and data requirements, secondly, are the most frequently described by software developer as an important aspect of developing a software application to support the adoption of EHRS. The latter implies that functional and data requirements are the most relevant for implementing the four cognitive processes. One possible conclusion is that functional requirements denote what the software should do, while data requirements specify characteristics of the required data. Then, as scenarios describe information services provided by an EHRS and the way in which physicians consume these services, software developers would give more attention to analyze the services than data. However, both the two are defined by software developers as important for implementing the interaction scenarios.

Otherwise, the less cited requirement for implementing the four cognitive processes is the context of use requirement. This would be explained because of interaction scenarios are a context description itself. Then, in some extend the software developers would discriminate context requirements because of them are implicit in the interaction scenario description.

By the other hand, Table 1 shows that usability goals are the second less mentioned requirement for implementing the four cognitive processes. One potential explanation is that the interaction scenarios do not explicitly describe usability goals, e.g. efficiency, effectiveness, and safety. Then, software developers would consider less important this kind of requirement.

As it is observed in previously described results, our findings are strongly influenced by the scenario based methodology. However, as this methodology is widely accepted and validated for gathering requirements [20], our results are an important step in the confirmation of functional requirements as a significant aspect for providing automatic support to adopt innovations in an EHRS.

Besides, when functional and data requirements are prevalent, usability goals and context of use requirements were considered less important by software developers (see Table 1). Thereby, these cases took place when developers examined contextualize and analyze user interaction cognitive processes. As Figure 8 shows, in a use case view, these results would be associated with the orientation of cognitive processes: Contextualize and Analyze user interaction processes are related with usage behavior; by the other hand, establish social network and Promote self-learning processes are related to the modification of intention to use an EHRS. In this sense, developers would associate functional and data requirements with usage behavior, and moreover, establish social network and promote self-learning processes would be associa-

ted with intention to use. These assumptions require more study, but would imply important considerations on the development of automatic support for assisting the adoption process of EHRS.

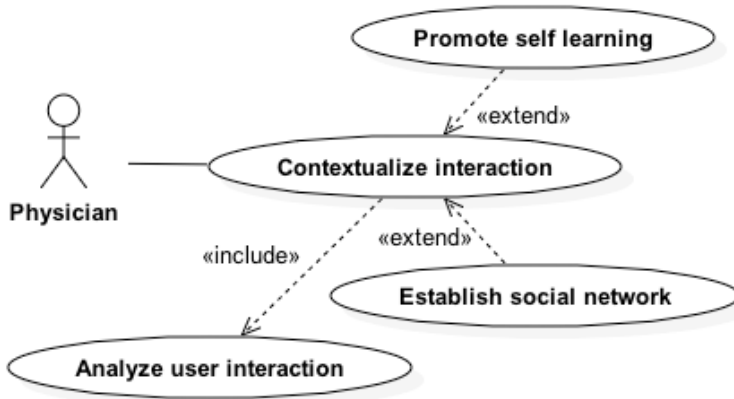


Figure 8. The cognitive processes from a view of system services.

Software requirement prioritization is an important problem in software engineering, and it has been studied from several perspectives. Some authors proposed approaches such as system case-based reasoning and neural network [21], case-based ranking [22], or software reliability growth modeling [23]. However, considering that the health care domain has a very particular culture, and adoption of EHRS by physicians is strongly influenced by socio-technical aspects, this study has taken a step in the direction of defining functional and data requirements as the most important for providing automated support to adopt innovations in EHRS. In addition, it is important to emphasize that sample size in the research design would limit our interpretations, nevertheless, the experience of the participants in the assessment sessions is an aspect that can give value to our results.

## 4. Conclusion

The adoption of EHRS is an important problem for providing health care services. Mainly, this problem has been faced from a user perspective, omitting the point of view of software engineers. The software engineer's point of view is important because of they play a central role as stakeholders in the development process of EHRS. In this chapter, we describe the importance of functional and data requirements for developing automated support to adopt EHRS. The prioritization of these requirements is supported from a perspective of software engineers. We believe our results are a

step in the provision of quality health care services, however we recommend that the approach outlined in this study be replicated in developers using a different development methodology to scenario based.

## 5. References

- [1] S. R. Simon, C. S. Soran, R. Kaushal, C. A. Jenter, L. A. Volk, E. Burdick, et al., "Physicians' Use of Key Functions in Electronic Health Records from 2005 to 2007: A Statewide Survey," *Journal of the American Medical Informatics Association*, vol. 16, pp. 465-470, 2009.
- [2] OpenMRS. (2015, January 18). OpenMRS Platform. Available: <http://openmrs.org/> (review font).
- [3] openEHR. (2008, February 5). The openEHR Foundation. Available: <http://www.openehr.org/>
- [4] V. H. Castillo, A. I. Martínez-García, L. Soriano-Equigua, J. L. Álvarez-Flores, and E. S. M. E., "Requirements prioritization to develop automated support for adopting EHRS," in 2016 4th International Conference in Software Engineering Research and Innovation, Puebla, México, 2016, pp. 9-14.
- [5] V. Castillo, A. Martinez-Garcia, and J. R. G. Pulido, "A knowledge-based taxonomy of critical factors for adopting electronic health record systems by physicians: a systematic literature review," *BMC Medical Informatics and Decision Making*, vol. 10, p. 60, 2010.
- [6] H. Sharp, Y. Rogers, and J. Preece, *Interaction design: beyond human-computer interaction*, 2d ed. West Sussex, UK: Wiley, 2007.
- [7] S. Collins, A. C. Hurley, F. Y. Chang, A. R. Illa, A. Benoit, S. Laperle, et al., "Content and functional specifications for a standards-based multidisciplinary rounding tool to maintain continuity across acute and critical care," *Journal of the American Medical Informatics Association*, vol. 21, pp. 438-447, 2014.
- [8] J. E. Hernández-Ávila, L. S. Palacio-Mejía, A. Lara-Esqueda, E. Silvestre, M. Agudelo-Botero, M. L. Diana, et al., "Assessing the process of designing and implementing electronic health records in a statewide public health system: the case of Colima, Mexico," *Journal of the American Medical Informatics Association : JAMIA*, vol. 20, pp. 238-244, 2013.
- [9] J. A. Osherooff, J. M. Teich, B. Middleton, E. B. Steen, A. Wright, and D. E. Detmer, "A Roadmap for National Action on Clinical Decision Support," *Journal of the American Medical Informatics Association*, vol. 14, pp. 141-145, 2007.
- [10] K. Kawamoto and D. F. Lobach, "Proposal for Fulfilling Strategic Objectives of the U.S. Roadmap for National Action on Decision Support through a Service-

- oriented Architecture Leveraging HL7 Services,” *Journal of the American Medical Informatics Association*, vol. 14, pp. 146-155, 2007.
- [11] G. J. Kuperman, J. S. Blair, R. A. Franck, S. Devaraj, and A. F. H. Low, “Developing data content specifications for the Nationwide Health Information Network Trial Implementations,” *Journal of the American Medical Informatics Association*, vol. 17, pp. 6-12, 2010.
  - [12] L. A. Lenert, D. Kirsh, W. G. Griswold, C. Buono, J. Lyon, R. Rao, et al., “Design and evaluation of a wireless electronic health records system for field care in mass casualty settings,” *Journal of the American Medical Informatics Association*, vol. 18, pp. 842-852, 2011.
  - [13] A. Muñoz, R. Somolinos, M. Pascual, J. A. Fragua, M. A. González, J. L. Monteagudo, et al., “Proof-of-concept Design and Development of an EN13606-based Electronic Health Care Record Service,” *Journal of the American Medical Informatics Association : JAMIA*, vol. 14, pp. 118-129, Jan-Feb 2007.
  - [14] W. V. Sujansky, J. M. Overhage, S. Chang, J. Frohlich, and S. A. Faus, “The Development of a Highly Constrained Health Level 7 Implementation Guide to Facilitate Electronic Laboratory Reporting to Ambulatory Electronic Health Record Systems,” *Journal of the American Medical Informatics Association*, vol. 16, pp. 285-290, 2009.
  - [15] V. Castillo and A. I. Martínez, “A knowledge management architecture for supporting the adoption of clinical information systems,” in *Eighth Mexican Int. Conf. on Computer Science (ENC 2007)*, Morelia, México, 2007.
  - [16] E. Rogers, *Diffusion of innovations*, 5th ed. New York: Free Press, 2003.
  - [17] C. Roda, A. Angehrn, T. Nabeth, and L. Razmerita, “Using conversational agents to support the adoption of knowledge sharing practices,” *Interacting with Computers*, vol. 15, p. 57, 2003.
  - [18] B. M. Ayyub, *A Practical Guide on Conducting Expert-Opinion Elicitation of Probabilities and Consequences for Corps Facilities*. VA, USA: Institute for Water Resources, 2001.
  - [19] M. Ali Babar and B. Kitchenham, “Assessment of a Framework for Comparing Software Architecture Analysis Methods,” presented at the 11th International Conference on Evaluation and Assessment in Software Engineering (EASE), Keele, Staffordshire, UK, 2007.
  - [20] M. B. Rosson and J. M. Carroll, *Usability engineering: scenario-based development of human computer interaction*, 1st ed. San Francisco, CA: Morgan Kaufmann, 2002.
  - [21] H. Mat Jani and A. B. M. Tariqul Islam, “A framework of software requirements quality analysis system using case-based reasoning and Neural Network,” in *In-*

formation Science and Service Science and Data Mining (ISSDM), 2012 6th International Conference on New Trends in, 2012, pp. 152-157.

- [22] A. Perini, A. Susi, and P. Avesani, "A Machine Learning Approach to Software Requirements Prioritization," *Software Engineering, IEEE Transactions on*, vol. 39, pp. 445-461, 2013.
- [23] P. L. Li, R. Nakagawa, and R. Montroy, "Estimating the Quality of Widely Used Software Products Using Software Reliability Growth Modeling: Case Study of an IBM Federated Database Project," in *Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on*, 2007, pp. 452-454.

# Chapter # 10

## Towards an Enterprise Architecture framework for an IT SME: Miracle Business Network

Maria-Isabel Crescencio-Lucero\*‡, Juan-Manuel Muñoz-Pérez\*‡, Alberto Portilla\*‡, Carolina-Rocío Sánchez-Pérez\*‡, Francisco Hernández-Jiménez\*† and Marva-Angelica Mora-Lumbreras\*

\* Autonomous University of Tlaxcala (UATx)  
Faculty of Basic Sciences and Engineering and Technology  
Apizaquito S/N Street, Apizaco, Tlaxcala

‡ Miracle Business Network S.A. de C.V. (MBN)  
Innovation and Development Center (CIDT)  
37 Street, No.216 La Loma Xicohtencatl Tlaxcala, Tlaxcala

† Higher Technological Institute of the Sierra Norte de Puebla  
José Luis Martínez Vazquez Avenue No. 2000, Jicolapa, Zacatlan, Puebla  
fmaria.isabel.c.l, juan.manuel.mp8, alberto.portilla, krlinasp, franherjim, marva.morag@gmail.com

### 1. Introduction

Nowadays most of the companies in Mexico are micro, small or medium enterprises, making up more than 95% of the total industry, which makes them an essential part of the Mexican economy [1]. In this context, the sector dedicated to Information Technologies (IT) shows sustained growth and is becoming increasingly important in the country's economy. In 2010 the IT sector represented 5.6% of gross domestic product (GDP) and its importance was greater than the total of primary activities (3.8%) [2]. In [3] it is emphasized that Small and Medium Enterprises (SMEs) IT have certain limitations to achieve competitive performance at national and international level. The problems they face are diverse, including the following: access to finance, skills weak managerial and labor management, lack of market opportunities, new technologies and methods of work organization. As a result, many SMEs do not show sustained growth and experience higher operating costs and higher rates of business failure. An Enterprise Architecture (EA) is a way to establish methodologically the elements through the time associated with the ITs that form part of the capital of the company.

It allows you to align and integrate dynamically objectives and business processes of an organization with the information technology required to, i) guide, limit and prioritize the plan investments, ii) justify the application of emerging technologies, iii) improve communication among all areas of a given company and, iv) provide a structure that displays the complexity and interaction of systems and applications. This is why companies whose adequately define its own EA are better prepared for decision-making, adaptability to the environment and support the growth of it.

There are several frameworks to define an EA, such as Zachman, TOGAF, FEA, and DODAF, which provide a series of steps to reach their definition, implementation and application within an organization. For the implementation of a business process architecture you can choose one or more frameworks. However, each framework has components that differ from each other, therefore it is necessary to define a set of characteristics according to the needs and objectives of the company in order to define which one will be used. The framework chosen should be able to provide tools focused on the design, planning, implementation and governance, these aspects enable the continuous improvement of the organization over time. In this chapter, we aim to define the dimensions and features that are useful for an SME business, and in particular for Miracle Business Network (MBN), which is an IT SMEs in the Mexican market. We compare the TOGAF, FEA and Zachman frameworks in order to obtain an ad-hoc framework for MBN needs. It is an extension to published paper at the 4th edition of the International Conference on Software Engineering Research and Innovation (CONISOFT'16), on this chapter have been added the implementation proposal section and another future works [14]. The rest of the chapter is organized as follows, section II introduces the concept of EA and a few representative EA, section III summarizes the methodology proposed for the election of the framework in an SME and the way it was working, section IV shows you how to use our work taking criteria for major relevance for SMEs MBN, section V contains works related to ours, section VI describes the proposal implementation in MBN Company and some diagrams that have already been designed related with the MBN Enterprise architecture and finally in section VI the conclusions that were reached with this work and future work are detailed.

## 2. Enterprise Architecture

An EA can be seen as a framework for the definition of the information involved in business processes of the company and its support by using IT. An EA framework defines the alignment between business strategy and information technology [4, 5, 9].

The Open Group Architecture Framework (TOGAF) defines the term Enterprise Architecture as [5]:

- A formal description of the system detailed at the component level to guide its implementation plan.
- The structure of the components, their interrelationships, and the principles and guidelines governing their design and evolution over time.

An EA is related to i) the goals of an organization and ii) how the systems can propose ways to organize processes in order to optimize resources and achieve objectives. Six components are identified within an EA: strategy, IT governance, information systems, information technology services, use and appropriation. Because of there are several methodologies and frameworks that give guidelines on how to build an EA, in the next subsections, we present a brief description of most interesting EA frameworks for this chapter.

## 2.1 TOGAF

TOGAF (The Open Group Architecture Framework) has a set of basic guidelines for the analysis and definition of the status of an EA. Specifically, it presents a method for developing EAs called ADM (Architecture Development Method) [11]. TOGAF is based on four dimensions:

- Business Architecture (or Business Process), which defines the business strategy, governance, structure and key processes of the organization.
- Application Architecture, which provides a plan (blueprint) for each application systems required to implement, interactions between these systems and their relationships with core business processes of the organization.
- Data Architecture, which describes the structure of the physical and logical organization data, and management resources of these data.
- Technology architecture, which describes the structure of hardware, software and networks required to support the implementation of the main applications of the organization.

The main TOGAF objectives is to establish a link between Business and IT companies, providing multiple benefits to both areas, such as cost reduction, risk reduction, identifying opportunities, flexibility, adaptation and common language.



## 2.2 ZACHMAN

It is a framework for EA, created and supported by ZIFA (Zachman Institute for Framework Advancement). This framework uses models and views of the different elements that are part of the EA, considering two dimensions [10]:

- Participants perspectives or models.
- Basic questions or views.

The framework defines the artifacts that are part of the architecture, employing a common language for all involved. It serves basically for implementing EA in companies (large, medium or small). To carry out this task of definition and implementation, Zachman considers different profiles, roles and skills that should be involved in the process, and special emphasis on the problems of communication and understanding existing between these profiles. To achieve this understanding in a simple and intuitive way, Zachman defines the following questions, to be answered by each profile to define completely Architecture: What? - How? Where? - Who? - When? - Why? [9], each model is related to a particular profile within the company and are indicated by the scope, business, system, technology, detailed representation, and the configuration of components and functional levels of the company. Zachman contributes to the simplicity, flexibility, standardization and adaptability when an EA is defined.

## 2.3 FEAF

FEAF (The Federal Enterprise Architecture Framework) is used for the development, management, maintenance, and decision-making related to an EA. FEAF provides a structure to organize resources and to describe the process management of enterprise architectures [12]. It is a collection of correlated references models which help with the definition of business functions and the analysis and optimization of IT operations.

FEAF allows the integration of information and the sharing of such information, therefore, it helps for the establishment of collaboration between the organizations. FEAF contains five reference models:

- Business Reference Model (BRM)
- Data Reference Model (DRM)

- Application Reference Model (ARM)
- Infrastructure Reference Model (IRM)
- Security Reference Model (SRM)

FEAF is used by the United States government organizations.

TOGAF, ZACHMAN and FEAF have in common 4 areas:

- Government.
- Data.
- Business.
- Technical aspect.

However, each framework approaches them with different philosophies: i) Zachman is a taxonomy used to organize architectural artifacts (design documents, specifications and models), ii) TOGAF is oriented to well defined processes in everyone enterprise areas, and iii) FEAF is a methodology that facilitates the analysis and identification of duplicate investment, differences and opportunities to collaborate within and across agencies.

Therefore, we argued that it is necessary to analyze each framework to select and implement the best features of them to define a framework help us with SMEs in our country.

### 3. Characterising EA frameworks

In this section, we present a methodology for analyzing EA frameworks in order to propose a set of dimensions for analyzing the implementation of an EA. The proposed dimensions are based on the MBN needs, which not only wants to define its own EA but it wants to incorporate it as part of the practices offered in the custom development systems.

#### 3.1 Analytical methodology for EA implementation

The Figure 1 shows the methodology followed for analyzing the implementation of an enterprise architecture in a SME Company.

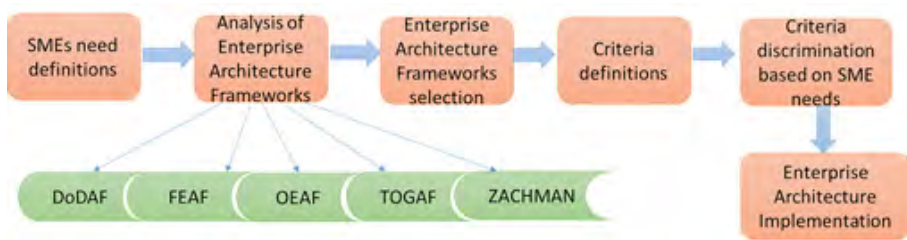


Figure1. Analysis methodology

- **Definition.** The definition of MBN needs at the moment of implementing an enterprise architecture. A key need for MBN is to define documents that contain the complete system architecture over different approaches, plus they must be aligned with the company objectives. The priority of an SME is to demonstrate that carries out its activities under quality standards but often they do not have enough budget to adopt international industry standards. In MBN there are certain problems in common with other SMEs (e.g. MBN Customers). The main problems and needs that they have are:
  - » There is a lot of information continuously increasing, so there is a need for managing such information to be useful.
  - » The technological infrastructure that MBN has is inefficient and / or employees unknown about its capacity. It must know the technology components owned by a company in order to manage and configure it, and to have the perspective that corresponds to address future problems with such technology components.
  - » MBN has systems for automating business processes, however, these are isolated efforts in particular areas without sharing information. Areas are requiring to provide information to achieve the companies objectives.
  - » The company MBN has processes based on standards like MOPROSOFT and CMMI. However, there are internal communication problems. Companies need to have a correct interaction between all processes.
  - » MBN does not have the financial resources to pay an implementation of an enterprise architecture framework but it has the need for structuring an EA by adopting or adapting frameworks characteristics that provide enough information.
- **Analysis.** It is based in a study of enterprise architecture frameworks; we have chosen 5:
  - » DoDAF Department of Defense Architecture Framework.

- » FEAF Federal Enterprise Architecture Framework.
- » OEAF Oracle Enterprise Architecture Framework
- » TOGAF The Open Group Architecture Framework
- » ZACHMAN

This selection is based on frameworks popularity and the organization activity to which they are focused.

- **Selection.** Three frameworks were selected (FEAF, TOGAF and Zachman) due to following reasons:
  - » Information available through different media, such as books provided by the framework creator, architecture books and scientific papers.
  - » They are among the most used frameworks in the enterprise market, except FEAF because it is focused on government area. However, it is important for companies to know what the government demands in the projects.
  - » The three Enterprise Architecture frameworks define the same type of architecture: business architecture, data architecture, application architecture and system architecture.

OEAF and DoDAF were excluded due to the following reasons:

- Missing documentation to perform an analysis of their operation
- They are not found within the Enterprise architecture frameworks of enterprise architectures commonly used in business.
- **Criteria definition.** A list of criteria was defined because of analyzing the needs obtained in the first stage. We conducted meetings with fourteen operating engineers, two process managers and two Administrative of MBN company. Besides, it is intended that the criteria list can help to SMEs for choosing an EA framework or for choosing certain criteria for each framework that are relevant to the company. The Table 1 shows the criteria definition we proposed with a discussion about its relevance in SMEs. Subsequently, an investigation was performed between frameworks to know if they can meet each one of the criteria. The result of this research was used to adjusting a framework for a company having clear their priorities and needs or to make the decision to implement its own EA framework. The Table 1 contains the result of the comparison of frameworks they are based on the 14 essential criteria for choosing the EA framework.

**Table 1. Criteria and their importance**

Criteria	Importance
Documentation access	To know the information availability in books and internet to analyze the enterprise architecture framework or to know if it is necessary to pay for get information.
Does it have certification?	It allows the company to be recognized by an accrediting body by the correct implementation of framework in the company.
How many layers or views it handles?	Defining the number of layers or views allow us to know the different types of architectural design in each software development project.
Does it define artifacts?	It helps the company to know the number, type and content of artifacts to generate by each layer or view architecture.
Does it have development method?	If the framework has an architecture development, method it will guide to the company through a set of steps to implement an enterprise architecture.
Does it describe what to implement?	It allows the company to know activities to be undertaken after to finalize each of the steps shown in the architecture development method.
Does it describe how to implement?	It provides a steps guide about how to implement the enterprise architecture framework.
Does it consider the business perspectives?	It will allow to show documentation focused on the role of the delivered area.
Target company size	Based on the company current size this allow us to know if it can be easily adapted within it.
Does it indicate how to classify the different artifacts?	It allows organizing the artifacts focused by areas.
Does it provide an evaluation of efficiency and maturity in the architecture implementation?	It lets to know about the implementation level that has a framework in the company, like evaluations CMMI or MOPROSOFT.
Does it provide a guide to understanding and creating a model of governance?	It will help take control of each of the products generated by the company for future changes and re-use, to gradually reduce response times of requested services.
Implementation time in months	Based on the need and urgency level it will allow the company to know the implementation time of each framework to make a decision about what framework must choose.
The complexity degree in the enterprise Adaptation	It displays if is necessary to do many changes in the actual enterprise process to implement the framework.

- **Criteria discrimination.** Based in Table 1 and Table 2 the discrimination criteria are done by assigning a value of important or unimportant, this importance value should be assigned by the staff of SMEs due to each has different needs.

**Table 2. Selection criteria**

CRITERIA	ZACHMAN	TOGAF	FEAF
Documentation access	Yes	Yes	Yes
Does it have certification?	Yes	Yes	Yes
How many layers or views it handles?	5	4	5
Does it define artifacts?	Not	Yes	Yes
Does it have development method?	Not	Yes	Yes
Does it describe what to implement?	Yes	Yes	Yes
Does it describe how to implement?	Not	Yes	Yes
Does it consider the business perspectives?	Yes	Yes	Yes
Target company size	Micro bigger or	Micro bigger or	Government
Does it indicate how to classify the different artifacts?	Good	Good	Excellent
Does it provide an evaluation of efficiency and maturity in the architecture implementation?	Not	Not	Not
Does it provide a guide to understanding and creating a model of governance?	Not	Yes	Yes
Implementation time in months	1 to 3	6 to 9	6 to 9
The complexity degree in the enterprise Adaptation	Medium	High	High

- **Implementation.** We compare how is implemented the criteria by each framework. We try to find the best practices by EA framework if possible. Otherwise, we combine best practices of several frameworks. After this selection, an implementation was performed in a real project in MBN Company.

## 4. Study Case: MBN

Once we have defined the general criteria for EA frameworks, a level of importance for each one according to MBN was assigned, in order to consider only those that align with the strategic objectives and needs of the organization.

For each of the 14 criteria were assigned two values of importance for the company: Important and Unimportant.

The Table 3 shows the criteria list with their importance for MBN Company, these values allow considering or not some aspects of each framework, which can later implement in an ad-hoc framework.

**Table 3. MBN Importance**

CRITERIA	MBN IMPORTANCE
Documentation access	Important
Does it have certification?	No important
How many layers or views it handles?	Important
Does it define artifacts?	Important
Does it have development method?	Important
Does it describe what to implement?	Important
Does it describe how to implement?	Important
Does it consider the business perspectives?	Important
Target company size	No important
Does it indicate how to classify the different artifacts?	Important
Does it provide an evaluation of efficiency and maturity in the architecture implementation?	No important
Does it provide a guide to understanding and creating a model of governance?	Important
Implementation time in months	No important
The complexity degree in the enterprise Adaptation	No important

The value of Important and not Important was obtained derived from a meeting with 8 operating engineers, 2 administrators and 2 business processes managers all from MBN, these criteria were shown and explained and based on their needs they assigned the corresponding value. As you can see the criteria that are not important to the company are those that would cause a specific framework implementation, for example, certification, cost or implementation time and the criteria that are important to the company are those that we can use to combine the best features from every framework. The result will be a hybrid EA implementation, in this case, we can mention as examples the criteria: documentation, method development and governance model, among others.

If the company chooses to take some criteria of different EA frameworks it should implement a hybrid architecture, considering the converging points from each architecture types and diagrams that we must be generated in each of them, also considering development methods and governance models. After getting this information, the company must choose which aspects of each framework are relevant for implementation.

## 5. Related works

Several frameworks provide guidance for implementing an AE in an organization focused on different business types, which hinders the process of choosing the EA framework that best suits a given company.

In [6] a comparative of enterprise architecture frameworks is presented: Zachman, TOGAF, FEAF and Gartner. In this research, an study case is described. The study is related to a pharmacy chain and how the problem would be solved with each of the EA frameworks described. They defined 12 criteria for comparing and evaluating enterprise architecture methodologies. They were based on these criterions to provide a brief recommendation based on assigned score to each criteria in order to choose the methodology that best suits to the company. Our work is strongly inspired by this research but focused in the context of a SME.

In [7] a comparison between TOGAF and Zachman was performed. To make this comparison they studied each of the frameworks highlighting the objectives they seek and the main points he works each. In this research, they use the 12 criteria defined in the work [6] and they mention that not all of these criteria may be relevant to the organization and some will be more important than others. They suggested that Zachman taxonomy could be used as a complement to TOGAF.

In [8] a work plan is done to reduce the initial complexity of the adoption of TOGAF framework prioritizing and adapting its application in a knowledge base architecture maturity models. The goal is that the initial dip to TOGAF was less daunting to take the standard. They propose to carry out a work plan for the implementation of TOGAF, which is based on a maturity model with continuous improvement NASCIO. The maturity model has six levels, where companies without level have an immature implementation and they can progress through the levels until to reach a mature implementation (level 6). As results, they got a mapping each NASCIO elements towards TOGAF elements, simplifying implementation and considering some elements related to the maturity level that the enterprise has. They achieved to separate the artifacts by each level NASCIO. His work is theoretical because it was not possible to test that all TOGAF elements can be implemented in an organization.

## 6. Implementation proposal

We considered that for choosing an EA framework, we must take into account the next question: what are the company needs? it will help us to know which framework is the one that meets those needs.

The main criterion for the design and implementation of enterprise architecture will be the number of layers or views that manage, considering the layers of greater importance for the company, regardless of the framework to which they belong.



Initially, the proposal is to implement the architecture layers that are similar in the frameworks, starting from this point, the company will have a base to continue implementing the layers that are different between the frameworks and they consider useful within your organization.

The three frameworks we analyzed have four layers in common, which are technology, business, data and systems. Such layers are considered in our proposal for implementation.

In the next subsections, we describe an implementation example of each architecture layer. All that is shown do not cover every defined aspect by each layer, but it gives us an initial perspective about how to begin the implementation.

The Figure 2 shows the architecture types defined by every framework, it is highlighted with the same color the layers that are similar, for example the Data Architecture layer from TOGAF, the System Information Model layer from Zachman and the Data Reference Model layer from FEAF are highlighted in blue color because they are modeling the architecture related to data.

FRAMEWORK	NUMBER OF LAYERS	ARCHITECTURES
TOGAF	4	Business Architecture
		Data Architecture
		Application Architecture
		Technology Architecture
ZACHMAN	6	Goals
		Business Model
		System Information Model
		Technology Model
		Detailed Representation
		Functional System
FEAF	5	Business Reference Model
		Data Reference Model
		Application Reference Model
		Infrastructure Reference Model
		Security Reference Model

Figure 2. Comparative of frameworks layers

## 6.1 Business Architecture

The Business Architecture defines the business strategy, government, organization and key business processes [5].

The MBN processes are based in NMX-I-059-NYCE2011 (MoProSoft) which is a Mexican quality standard, therefore they were considered in the business architecture design and implementation.

For the business architecture description was designed BPMNs diagrams, they model the workflow of each enterprise department. At enterprise level were defined the next processes:

- Business management
- Business projects
- Business resources
- Project Management
- Human resources
- Assets, services and infrastructure
- Organization knowledge
- Development and software maintenance

The Development and software maintenance process was designed and implemented using the quality model CMMI DEV 2. The Figure 3 shows you an example of the BPMN diagram related to the designed processes and implemented for the business architecture. Every rectangle represents a process for the enterprise.

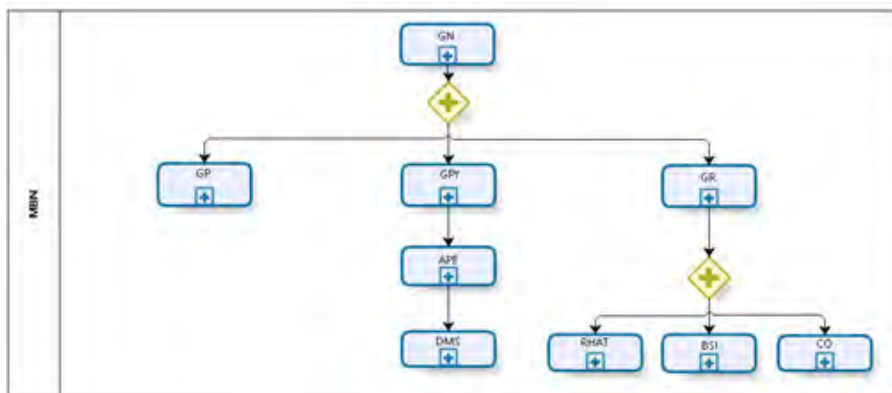


Figure 3. General diagram of business process

## 6.2 Technology Architecture

The objective of the technology architecture is to provide the IT support (platforms, operative systems, databases, network and telecommunications) [5]. The Technology Architecture design should be used as a guide when there is a need to work in a maintenance or upgrade of it (see Figure 4). The Technology Architecture document contains three sections:

- Devices On this section there are described the characteristics of the network components.
- Network diagram, this section includes a diagram that shows the enterprise topology network.
- Settings, it has the references toward another document that has instructions about network devices configuration.

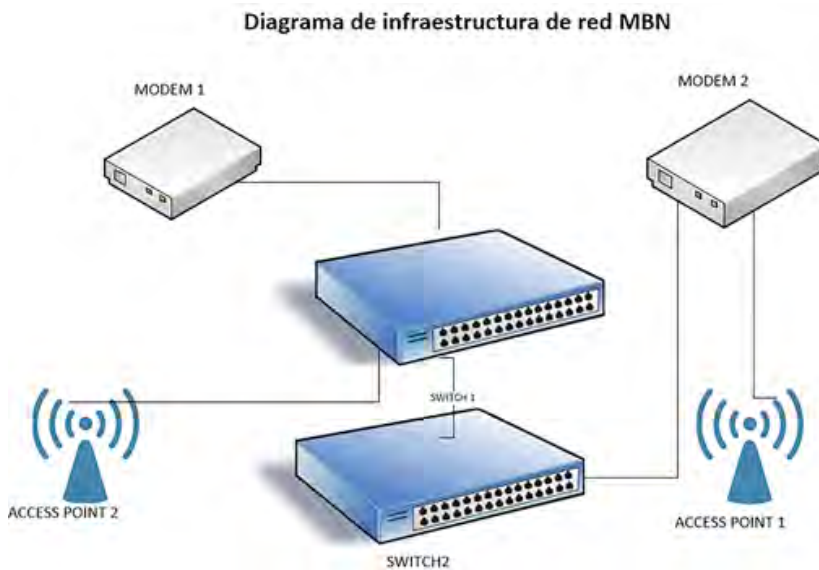


Figure 4. General Diagram of Network Infrastructure Company

## 6.3 System Architecture

The system architecture provides a plane for every system that will be implemented, interactions between them and their relationships with business process in the organization [5].

The objective is to show the design of system architecture implementation, their interactions and their relationships with the company business process. The diagram designed contains the systems and their users. The Figure 5 shows the diagram design; in it they can see the systems architecture in the company.

Nowadays the company has 6 systems and they are working by separate form, therefore they need to be integrated with a SOA Implementation.

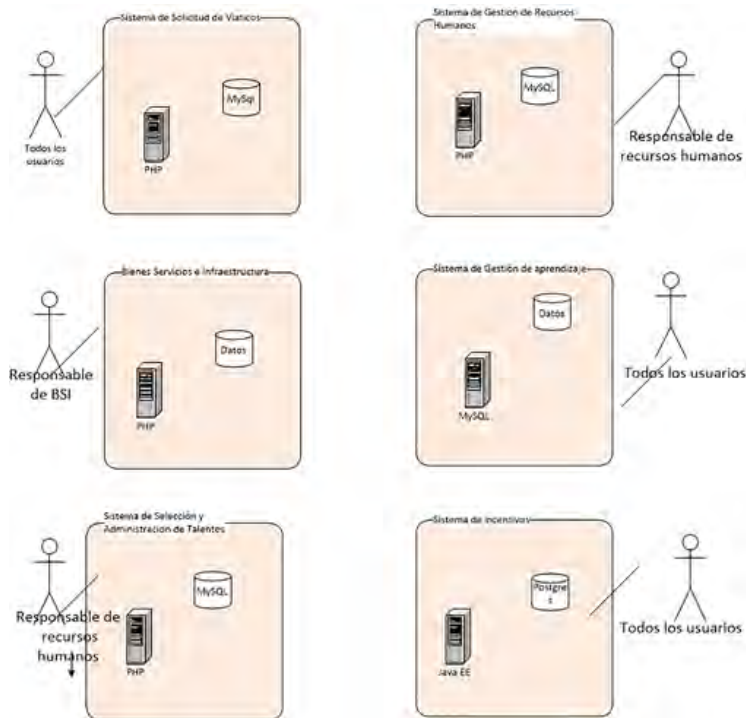


Figure 5. General Diagram of Systems Company

## 7. Data Architecture

The data architecture describes the logical organization enterprise and the management resources [5].

For this architecture was performed a design about how is management the enterprise information. To achieve this work a knowledge base was created and it contains every Enterprise area, their structure was defined by organization knowledge process because it is the responsible for add and remove permissions on it.

To manage the knowledge base, a document was created, it contains the general rules that must be followed when information is modified. Figure 6 show us the diagram design of this architecture.

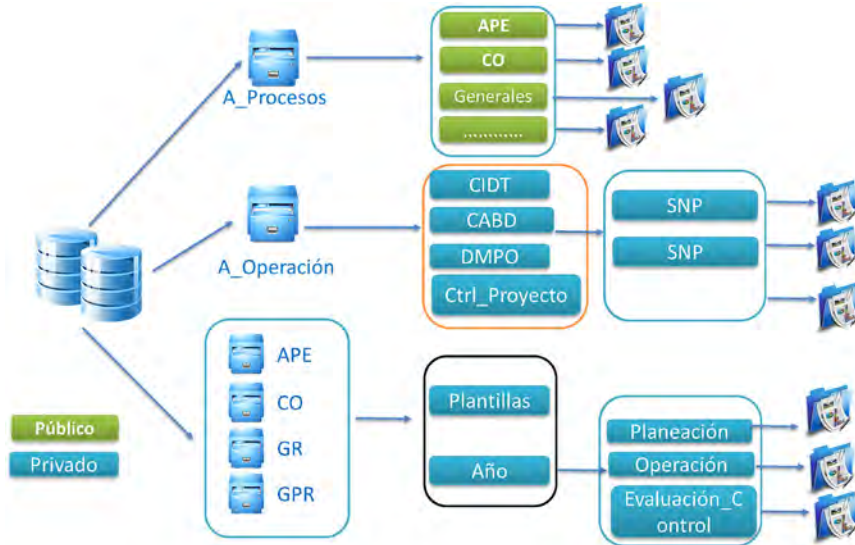


Figure 6. General Diagram of Data Company

## 8. Conclusions and future work

Today, organizations expect to have technological solutions that i) support the objectives and business processes, ii) manage IT services in an efficient way, and iii) address the company from its various dimensions (business, data, applications and infrastructure). For this proposal, process and methodologies have been designed to let to the organizations aligning their efforts to business goals under the described dimensions.

We argued that it is imperative that SMEs in our country are able to address issues related to best practices, methodologies, models and tools enabling them to function in a highly competitive, regulated and changing environment in order to meet the challenges that the IT industry demands. In this context, companies seek to define Enterprise Architectures that provide a model of integration supported by a strategic planning.

In this chapter, we present an analysis of three EA Frameworks and its associated criteria. It is important that companies wishing to implement an enterprise architecture

know the dimensions that can cover each of the frameworks. Based on the set of criteria defined in this chapter you can get a guide for choosing a particular framework or for performing an implementation of a hybrid EA within a given organization.

The analysis we present in this chapter will be the guide for the implementation of a hybrid enterprise architecture at the company MBN. Therefore, a practical proposal for the initial implementation of an EA was sketched.

We consider that for each criterion it is very important to follow the next steps:

- To perform an analysis of how the criterion is implemented in each of the frameworks chosen.
- To choose the framework that implements the criterion in the most complete, simple and understandable way, or trying to make a mixture of the best features provided by each framework.
- To perform the implementation of the criterion.

As a future work, we plan to present the implementation of the EA framework at MBN along with all the documents, artifacts, process and models related to such an EA framework. We are also working in a detailed way for defining a set of common needs of SME in order to propose an EA framework adapted to Mexican enterprises.

## 9. References

- [1] M.C. Gutierrez M.C., L. Pinon, A. Sap en. "Modelos de Calidad usados en PyMES de tecnologías de information ubicadas en el parque de Innovación y Transferencia de Tecnología de la Ciudad de Chihuahua. 2011.
- [2] R. Gallegos, C. Grandet, P. Ramirez. "Los Emprendedores de TIC en Mexico. Recomendaciones de política publica para su nacimiento, crecimiento y consolidación". 1ra. Edición. Instituto Mexicano para la Competitividad A.C. 2014.
- [3] G. Lopez-Acevedo y H. Tan. "Impact Evaluation of SME Programas in Latin America and Caribbean", Washington USA. World Bank. 2010.
- [4] A. Valdez-Menchaca, C. Vega-Lebrun, E. Olivares-Benitez,
- [5] J. Perez-García, O. Arzola-Garza. O. Preciado-Martínez, S. Castaneda-Alvarado. "Practical Application of Enterprise Architecture Study Case of SME Metalmechanic in Mexico", European Scientific Journal Special edition Vol. 1. 2013.
- [6] The Open Group, "Welcome to TOGAF® Version 9.1, an Open Group Standard", 2011. [Online]. Available: <http://pubs.opengroup.org/architecture/togaf9-doc/arch/>

- [7] Roger Sessions, ObjectWatch, Inc. "A Comparison of the Top Four Enterprise-Architecture Methodologies", 2007. [Online]. Available: <https://msdn.microsoft.com/en-US/enus/library/bb466232.aspx>
- [8] Alok Singh, Pankaj Mudholkar, Lovely Lakhmani Balani. "Contemporary Enterprise Architecture Frameworks (A Comparative study of TOGAF and Zachmans EA frameworks)" 2014
- [9] R.W. Gosselt, A Maturity. "Model based Roadmap for Implementing TOGAF", 17th Twente Student Conference on IT, 2012
- [10] John A. Zachman. "Architecture is Architecture is Architecture", 2011. [Online]. Available: <http://www.zachman.com/ea-articles-reference/52architecture-is-architecture-is-architecture-by-john-a-zachmant>
- [11] John A. Zachman. "What is Enterprise Architecture and why do we do it?", 2015. [Online]. Available: <http://www.zachman.com/faqs/20enterprise-architecture-faqs>
- [12] Philippe Desfray, Gilbert Raymond. "Modeling Enterprise Architecture with TOGAF", 2014
- [13] www.whitehouse.gov, "Federal Enterprise Architecture Framework Version 2", 2013. [Online]. Available: [https://www.whitehouse.gov/sites/default/files/omb/assets/egov\\_docs/fea\\_v2.pdf](https://www.whitehouse.gov/sites/default/files/omb/assets/egov_docs/fea_v2.pdf)
- [14] <http://dof.gob.mx>, "Diario Oficial de la Federacion", 2016, [Online]. Available: [http://dof.gob.mx/nota\\_detalle\\_popup.php?codigo=5096849](http://dof.gob.mx/nota_detalle_popup.php?codigo=5096849)
- [15] Maria-Isabel Crescencio-Lucero, Juan-Manuel Munoz-Perez, Alberto Portilla, Carolina-Rocio Sanchez-Perez, Francisco Hernandez-Jimenez and Marva-Angelica Mora-Lumbreras "4to. Congreso Internacional de Investigación e Innovación en Ingeniería de Software". Tijuana, Baja California: Universidad Autónoma de Baja California, 2016. ISBN 978-0-692-69638-5 p. 23.

# Chapter # 11

## User-oriented application for source code metrics definition and extraction based on a metrics framework

Alberto S. Núñez-Varela, Héctor G. Pérez-González, Francisco E. Martínez-Pérez,  
Juan Cuevas-Tello  
Facultad de Ingeniería  
Universidad Autónoma de San Luis Potosí  
San Luis Potosí, México  
alberto\_snv@hotmail.com, hecторgerardo@yahoo.com, fcoemtz@gmail.com,  
cuevas@uaslp.mx

### 1. Introduction

Software product metrics are essential tools for the metric measurement process. A metric is defined by the IEEE Std. 610.12-1990 standard [1] as “A function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which software possesses a given attribute that affects its quality”.

Metrics are usually divided on two categories, process and product metrics [2]. In this research, we are interested in product metrics for source code measurement, which are metrics used to measure quality attributes of the source code such as size, complexity, cohesion, coupling, design, maintainability, etc. These metrics are usually known as source code metrics and are the main focus of this paper. These types of metrics include well-known metrics such as lines of code (LOC), cyclomatic complexity [3] and the Halstead metrics [4], along with object-oriented metrics such as the Chidamber and Kemerer metric set [5] and the MOOD set [6]. Metrics have been used over the years for measuring software quality attributes, but they provide a wider range of applications and studies, such as fault prediction and defect prediction, as shown in [2].

In order to extract the source code metrics, several mechanisms and tools have been proposed over the years, but the process for metric extraction is not an easy task and



as software systems grow in size, their components, such as the source code, become harder to handle. Systems with thousands of lines of code could make the metric extraction process an almost impossible task if done manually [7]. In order to handle source code of any size, automated tools have been developed for metric extraction, and these tools are usually known as Software Metric Tools (SMT). Many tools have been proposed through the years as research and commercial efforts, but due to the number of metrics and programming languages in existence, these tools are not easy to create and maintain.

Current software metric tools, especially if commercial, present some issues. The tools characteristics are affected and limited by the set of metrics defined and the number of programming languages these recognize; the user is not allowed to define new metrics or incorporate new languages into the tools. It is also important to mention that current software metric tools focus on object-oriented metrics, and other programming paradigms have been neglected. Object-oriented programming is the most common and used programming paradigm for software development, but other paradigms are widely used and studied. Two paradigms in particular, aspect-oriented programming and feature-oriented programming, have been gaining importance in the last years, this because of the current interest in programming concerns, software product lines and big scale software. Different paradigms introduce particular characteristics to the code and new quality attributes to measure, because of that, new code metrics must be defined in order to measure the particularities of the source code. If a tool does not allow the incorporation of new languages, it will impact the support of new paradigms. Next to these limitations, researchers have identified several related issues with metric tools; these issues are discussed in Section 3.

Research efforts have been made in order to solve those issues in the form of metric frameworks. These frameworks propose algorithms, techniques and methods for source code metric extraction in a more generic way, that is, language and metric independent. A brief description of these frameworks is presented in Section 2. Based on a metric framework, this paper proposes the construction of a user-oriented tool, in which the tool will not limit the user by its capabilities, instead, the tool will allow the user to define the desired metrics and implement new programming languages. This proposal will be achieved by allowing the user to directly query the source code based on the language grammar definition.

A shorter version of the work presented here was presented in [30]. This new version provides details on the proposed tool architecture, comparison with the current

tools, as well as further details on the proposed methodology used for the metrics definition and extraction.

In the following section, related work about the topics discussed here is presented.

## 2. Related work

Software metric tools have been a subject of study for many years. They represent a research opportunity for many researchers given the difficulty and challenges they represent, and have been gaining major importance with the current research interest in new programming paradigms, such as the aspect and feature oriented programming. Research papers presenting software metric tools, and the methods used to define and extract the metrics, have been presented over the years. The tool CKJM presented in [8] is probably one of the most used and referenced tools in the research community. The tool is able to extract the Chidamber and Kemerer metrics from Java source code. Other research tools found in the literature are presented in Table 1. These tools are presented as research efforts and have specific uses. Each tool measures a predefined set of metrics for source code written in C/C++ or Java.

Table 1. Research tools

Tool	Defined in
CCMETRICS	Husein and Oxley [9]
CKJM	Spinellis [8]
OOMeter	Alghamdi et al. [10]
PROM (Pro Metrics)	M. Scotto et al. [2], Sillitti et al. [11]
QMOOD++	Bansiya and Davis [12]
TAC++	Fioravanti and Nesi [13]

These tools, along with commercial tools, present the limitations we have been discussing; they are dependent on the metrics and languages they are built for, and even if they share a common architecture, each tool works differently. Some tools use intermediate representations of the source code, databases for storing the extracted data from the code, and in the case of CKJM, it does not measure directly the source code, instead it measure Java bytecode. All these different characteristics and capacities for each tool represent a problem when choosing a tool for code metrics extraction. For some cases, a single tool will not provide enough characteristics to fulfill the measurement needs, thus, forcing the necessity of using more than one

tool for the metrics extraction process. The use, challenges and study of software metric tools have been reported by authors [14, 15, 16]. These challenges represent a major problem since software metric tools are essential for any measurement study, but researchers still do not trust entirely in the technology behind current tools, to the extent of opting for the development of their own complex and time consuming tools. These issues are discussed in Section 3.

As mentioned before, metrics frameworks have been developed in order to provide a functional base for a generic metric extraction process [17, 18, 19, 20, 21, 22, 23]. These frameworks propose mechanisms for metric extraction and can be classified into two groups. In the first group, we classify the frameworks that use intermediate representations of the source code or language grammar for their operations; data structures such as graphs or trees are usually used for this intermediate representation. In the second group, we classify the frameworks that use models to define and manipulate a representation of the source code; these models are usually based on standards such as the Unified Modeling Language (UML) [24] and the Object Constraint Language (OCL) [25]. Each group of frameworks provides different mechanisms with certain advantages and disadvantages; some are easier to use than others, and some, even if they are trying to achieve generic measurement, are directly linked to a certain programming paradigm.

Frameworks using UML models for metric definition and extraction are presented in [17, 20, 21, 22]. These type frameworks are aimed to object-oriented source code and their overall objective is to construct a UML meta-model of the source code identifying components and entities of interest, this meta-model is later manipulated via OCL or any other technique, in order to extract the desired information.

Frameworks based on intermediate representations are presented in [18, 19, 23]. In [19, 23] an intermediate graph representation of the source code is done, the graph is then manipulated in order to extract the results. In [18] a tree representation of the language grammar is presented in which each node can be queried in order to extract the metrics. This last framework is on ongoing work of the authors of this paper and is the framework that will be used as part of this work.

A relevant point of consideration is the facilities of use these frameworks can provide. This is important since we are looking to implement a framework as part of a tool, so it must provide some mechanism for user interaction. From the frameworks discussed, three of them [18, 20, 21] present this user interaction in the form of a query language.

Based on the framework presented in [18] we propose the characteristics and construction of a user-oriented metric tool. It is based on the definition and manipulation of a programming language grammar through queries, which are used to defined and extract the desired source code metrics.

The structure of the paper is as follows. In Section 3 the overall structure and problematic with current software metric tools are presented. Section 4 presents the proposal of a user-oriented metric tool. In Section 5 the results and future work are presented, and we present our conclusions in Section 6.

### 3. Current software metrics tools

Software Metric Tools (SMT's) are computer programs which aid the process of metric extraction. They are very important since the process of metric extraction is not an easy task, it is time-consuming, and is a process that might become virtually impossible to do it manually [7]. The current size of software systems, especially with the growing interest on big scale software and software product lines, the number of programming languages, and the number of quality attributes to measure, make the use of automated software metric tools applications a primary necessity in the software measurement process.

The overall behavior of a metric tool for metric extraction consists on parsing and extracting a set of metrics for the input source code. For doing that, the tool must implement and recognize a determined set of programming languages it can parse, along with a set of metrics it can process and ultimately extract, and because of that, the users of these tools are often limited by the characteristics of the tool. The purpose of this work is to provide the characteristics a metric tool should possess in order to be user oriented, meaning that the user is the one defining metrics and languages, not the tool.

Current software metric tools range from research tools to commercial tools. In Table 1 we provided a brief list of tools presented as research papers, and Table 2 provides a list of representative commercial tools, the main programming languages they support, and their website. It is not the objective of this work to present a study or description of the tools; we are interested in the general problematic these tools present.

#### 3.1 Common architecture

Metrics tools, especially if commercial, are usually black boxes and present a simple architecture. The user inputs a set of source code files written in a language the tool recognizes, along with a selection of metrics the tool provides, and the tool outputs the

extracted metrics. Little interaction is permitted between the user and the tool, thus limiting the user to work with what is already defined, directly affecting the necessities of a measurement project.

**Table 2. Commercial tools**

Tool	Languages	Website
Understand	C, C++, C#, Java	<a href="https://scitools.com/">https://scitools.com/</a>
Krakatau Professional	C++, Java	<a href="http://www.powersoftware.com">http://www.powersoftware.com</a>
JHawk	Java	<a href="http://www.virtualmachinery.com">http://www.virtualmachinery.com</a>
McCabeIQ	C#, C++.NET, C++, JAVA, VB.NET	<a href="http://www.mccabe.com">http://www.mccabe.com</a>
Resource Standard Metrics	C++, C#, Java	<a href="http://msquaredtechnologies.com/">http://msquaredtechnologies.com/</a>
SDMetrics	From UML	<a href="http://www.sdmetrics.com/">http://www.sdmetrics.com/</a>

### 3.2 Tools problematic

Many authors have pointed out issues with software metric tools. These are not minor problems and truly affect the measurement process. The following issues comprise and summarize the major issues found with metric tools:

1. Tools are dependent on the set of metrics and languages they accept [18, 19, 26]. This is especially important since every project has their measurement needs and source code can be written in different programming languages. In addition, new metrics and programming languages can be proposed, or modifications can be made to the existing ones. This is a very important issue since it restricts the measurement needs of a particular study; especially for those exploring not widely used metrics, or those proposing new metrics.
2. Tools cannot be extended to accept new metrics or languages [18, 19, 22, 27]. Current metric tools do not usually provide mechanisms, or easy mechanisms, that could allow the user to escalate the tool's definitions; the user is left with the original functionality of the tool. Tools like Understand and JHawk allow the definition of new metrics from the already predefined data extracted from the source code, but no real new metrics definitions are accepted, only modifications to existing ones.
3. Different and inconsistent results across tools [13, 16, 18, 22, 26, 28]. In [28] a study is made in which the authors compare the results for the same metric obtained in different tools and conclude that the results are different. Each tool can provide a different interpretation and consequently a different imple-

mentation of each metric, this according to the understating and knowledge of the developers of the tool. This represents a major issue since metrics empirical studies cannot be easily reproduced by researchers if different results are obtained according to the tool.

4. Updates of the tool must be made according to languages changes [14, 29]. Even though languages changes are not frequent, these changes usually represent major improvements in the language, including new characteristics and syntax that are not recognized by default by the tool. In fact, since syntax parsers are a key component in metrics tools, newer source files provided by the user might not be recognized at all by the parser. Also, dialects for major programming languages exist whose code might not be recognized by a certain tool.

Another issue is that they are not free of cost, which even if it is not a technological related issue, the cost might represent a major factor when choosing a tool. Even though most of the commercial tools provide a demo or trial version, the restrictions these versions contain impact the tool functionality to the point of not being entirely useful for a measurement project. Research tools might not present a viable solution for replacing commercial tools, and therefore eliminating the cost issue, since they are presented for specific goals, have limited availability and are not updated.

## 4. User oriented metric tool

We propose the characteristics and construction of a user-oriented metric tool, in which the main purpose is to allow the user to define metrics and languages.

### 4.1 Architecture

A similar architecture to the one defined in Section 3.1 is proposed for this tool. The main difference is that the black box is substituted by a framework and the user can define metrics and languages which will be processed by the framework. This will allow the user to provide the metrics and languages definitions as an input to the tool.

It is important to indicate the differences between a common metric tool and a user-oriented tool. Fig. 1 depicts the architecture of common software metrics tools, and as it can be observed, it is usually common to have a parser for each programming language accepted by the tool, with metrics usually being defined and computed by means of these parsers. This type of architecture is complex and hard to maintain and modify.

Some tools try to simplify and reduce the complexity by providing an intermediate representation of the source code that is common across languages as depicted in Figure 2. Therefore, instead of defining the metrics on the parsers, they are defined and computed by means of an intermediate representation of the source code. This intermediate representation is achieved by creating common structures such as graphs or trees, or by using standard formats such as XML. Even though using intermediate representations allows more flexibility, and modifications to the tool are easier (all languages recognize a metric modification), there is still a need of individual parsers in order to create those representations.

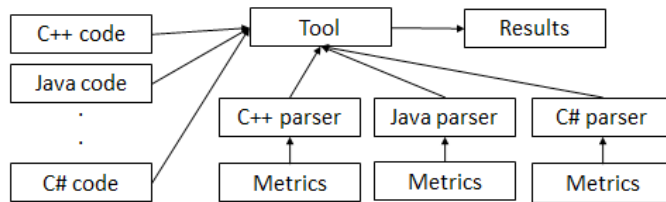


Figure 1. Common metrics tool architecture

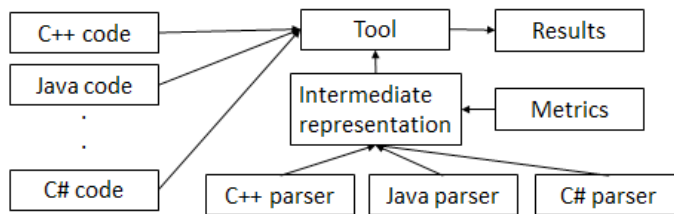


Figure 2. Common metrics tool architecture (simplified)

The user oriented tool architecture is depicted in Figure 3. The first goal, and main difference from the common tools, is the absence of parsers and intermediate representations. Intermediate representations are a good option for providing a common platform when processing different languages, but their use might add a layer of complexity. On the other hand, individual parsers are removed and all the language processing is then delegated to a framework with mechanisms for language processing and metrics extraction. The framework will allow the individual working with the tool to provide a programming language grammar as an input, as well as the metrics definitions he wants to extract. User defined languages and metrics is a characteristic that is not easily achieved by a common architecture, this is because a parser must be provided in order to accept the language, and the metrics are sometimes hard coded in the parser or tool code.

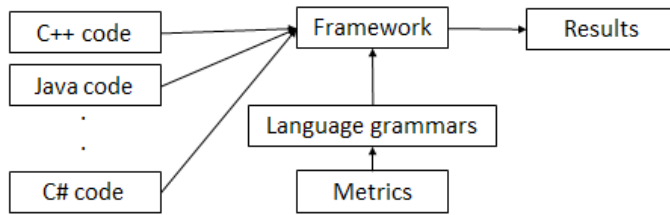


Figure 3. User-oriented tool architecture

## 4.2 Implementation

The main goal for the implementation of a tool as proposed here lies entirely in the framework or methodology that could provide a generic metric extraction. In Section 2, we described frameworks that have been proposed in order to achieve that, but might not be entirely useful as a metrics tool base. For this purpose, we have been developing a methodology which will serve as the base for a user-oriented software metric tool. A preliminary report and results of this methodology have already been published in [18]. Next, we describe the methodology as implemented in the construction of the tool.

The methodology aims to obtain a set of arranged substrings from the source code regardless of the programming language the source code is written. These substrings are processed in a way that will have a meaning to the user. This process will be achieved, by allowing the user to query the source code directly and the results of those queries will be the set of aforementioned substrings.

The starting point of the methodology is the definition of the language. A programming language is usually defined by a context free grammar, which defines the syntactical and lexical structure of the language. From the grammar, an intermediate tree type representation of the grammar is created, and we call this representation a Path Tree. From this representation the queries can be performed and the results obtained.

A Path Tree is an intermediate tree representation of the language grammar which could be compared to an abstract syntax tree (AST), but with major differences. The path tree is created statically as a pre-processing step, unlike the AST that is created at derivation time. The path tree is created in order to obtain a more manageable representation of the grammar to be queried and store the results. Given the nature of the grammar itself, recursion and ambiguity in the creation of the tree represent two problems to solve. Recursion is presented in a production of the type  $A \rightarrow A\alpha$ , which would create infinite loops in the parent-children relations, and ambiguity is presented when there is more than one



way from which a terminal can be derived. In order to prevent that, we define the way in which the tree must be created.

The nodes of the tree are created from the non-terminals symbols of the grammar and will define all valid paths that can be queried. Each node is labeled with a non-terminal symbol following the next rules from [18]:

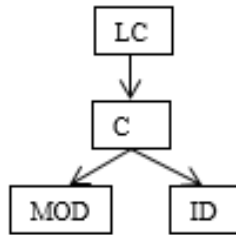
- The root node of the tree is labeled with the symbol found on the left side of the initial production.
- For each node N labeled with the symbol S is due to meet to:
  - » The labels of child nodes of N may not be repeated and are different from S.
  - » In the branch that is built from node N, node N can only be labeled with the symbol S.
- Valid paths are those that are formed from a top - down tour from any node to any other node. If the paths begin to form from the root of the tree (which is the start symbol by rule 1) does not get any ambiguous path.

When the actual parsing of the source code happens, every node (non-terminal) of the tree will be filled with their corresponding lexical values (tokens). For more information on this topic, please refer to [18].

From this point, we need to allow the user to query this tree. This is important since it provides the interaction and user metric definition we are looking for. From the tree, each non-terminal can be queried so the user will have to simply input the name of the non-terminal to query. As mentioned above ambiguity can be presented since a non-terminal can be derived from different paths, so for a more generalized representation, we are going to define a query from the tree root to the desired node. We are going to use the symbol “!” in order to denote parent-children relationships. As an example, we provide the following grammar:

$LC \rightarrow C$   
 $LC \rightarrow C LC$   
 $C \rightarrow MOD(\_ \wedge) class \wedge ID \{ \wedge \}$   
 $MOD \rightarrow 'public' | (\_ \wedge) private \wedge$   
 $ID \rightarrow identifier$

From the grammar, the following tree is created:



The language defined by the grammar accepts the following code fragment:

```

public class MyClass {}
private class MyOtherClass {}
  
```

The lexical values obtained in the parsing process are saved in the corresponding tree node (non-terminal). Table 3 shows each node and their corresponding values.

**Table 3. Node values**

Non-terminal	Values
LC	public class MyClass {} private class MyClass {}
C	public class MyClass {} public class MyOtherClass {}
MOD	public private
ID	MyClass MyOtherClass

Now we can create the queries from the tree. Table 4 presents two queries and the results.

**Table 4. Queries Examples**

Query	Meaning	Lexical result	Numeric result
LC!C!ID	Classes in the code	MyClass MyOtherClass	2
LC!C!MOD	Modifiers used in the code	Public Private	2

Querying single non-terminals provide good information and allow the extraction of common count metrics, such as “*number of classes*” defined in Table 4, but we also need to provide a level of specialization by linking results. Specialization is a common characteristic in code metrics which allows defining a metric with a certain scope, for example

the metrics “*number of methods*” or “*number of attributes*” are defined in the scope of a class. As an example, we present the following grammar which allows the declaration of classes with member variables:

$LC \rightarrow C$   
 $LC \rightarrow C LC$   
 $C \rightarrow MOD(\wedge)class\wedge ID \{ \wedge VARS \}$   
 $MOD \rightarrow 'public' | (\wedge)private\wedge$   
 $ID \rightarrow identifier$   
 $VARS \rightarrow VARDEC VARS | VARDEC$   
 $VARDEC \rightarrow MOD(\wedge)int\wedge ID$

The language defined by the grammar accepts the following code fragment:

```
public class MyClass
{
    public int var1, var2
    private int var3
}
private class MyOtherClass
{
    public int var4, var5
}
```

Given the grammar of the language and the code, we can query the terminals. For this example, we want to query the metric “*number of attributes*”, which extracts the member variables for each class. Table 5 presents the queries needed for the metric.

Table 5. Single queries

Query	Meaning	Lexical result
LC!C!ID	Classes	MyClass MyOtherClass
LC!C!VARS!VARDEC!ID	Member variables	var1 var2 var3 var4 var5

Each query provides the necessary lexical values needed by the metric, but they are provided as different results sets. We need to link both queries in order to get a single

result. For that, we merge the queries by linking them with a given operator; we use the symbol ‘&’ for this purpose. By linking the queries in Table 5 we get the query: `LC!C!ID & LC!C!VARS!VARDEC!ID`. The lexical values are also merged in order to get the result as presented in Table 6.

Table 6. Query merged result

LC!C!ID result	LC!C!VARS!VARDEC!ID result
MyClass	var1
	var2
	var3
MyOtherClass	var4
	var5

As mentioned above, the lexical values are merged and accommodated according to their real character positions in the source code.

### 4.3 Results of user interaction through queries and grammars

In this section we provide a summary of the results obtained by applying the proposed methodology in the construction of a user metrics oriented tool. As we are looking for a complete user interaction with the tool, the proposed framework is aimed to achieve that. Since the functioning of the framework is based on the language grammar, the user can provide any grammar to the tool and the functionality will remain the same, and thanks to that, the tool is not limited by the languages it was developed for.

Since the framework works by querying the grammar representation, those queries can be performed by the user and each query will represent a well-known metric, or any other measure the user would like to extract. By allowing the user to define the measures through queries, the tool is no limited by the set of metrics that it can extract.

We have tested our methodology measuring C# and Java code. We defined and generated a variety of metrics, including well-known metrics such as number of classes, methods per class, member variables, data types used, fan-in, cyclomatic complexity, etc. Next, we present a brief list of metrics we are able to extract trough queries:

- Namespaces used in each class
- Number/name of classes
- FanIn/FanOut
- Cyclomatic complexity
- Number of language instruction (vocabulary, conditions, cycles, enumerations, exceptions, etc.)
- Per class:
  - » Public, private and protected member variables.
  - » Public, private and protected methods.
  - » Parent classes.
  - » Interface complexity (parameters + return points).
  - » Class complexity (cyclomatic complexity + interface complexity).
  - » Data types.

Most of the listed metrics can also be extracted depending on the desired scope according to the programming language (namespace, package, class, method, etc.). The results are obtained from a simple application we built based on the proposed methodology. The application allows the user to input the queries from the defined grammar and a set of source files, and outputs the results in plain text or XML format.

The results obtained so far are promising and will allow us to build a complete overall user oriented application. This type of application can provide two layers of use depending on the user expertise. In one layer, the application itself will contain predefined metrics and programming languages from which any user can easily choose from, and in the other layer, advanced users will be able to define new metrics and incorporate new programming languages to the tool. Metrics tools for source code measurement are essential for the overall software measurement process, and a tool like the one proposed here, will be of great help for researchers and practitioners, especially for those researchers working on the definition and validation of new metrics not supported by current tools, since the time and effort required to conduct a metrics research study can be greatly decreased if no additional tools or analyzers have to be built as part of the research process.

## 5. Future work

The creation and correct functioning of a tool like the one we propose depends heavily on the framework and the interaction with the user. Using queries for user input is fundamen-

tal for the type of user interaction we are looking to achieve, and further work must be done in order to formally define a complete query language based on those queries. The query language will provide metrics relations, mathematical operations, conditions, etc.

This query language will allow us to extract metrics which require additional processing, for example, the metric *number of comments per method* requires the computation of two metrics. The framework itself is well suited for extracting the metrics, but aided by the query language we will be able to calculate the desired relation. This language will not require the framework to be modified because it only requires a post-processing of the results.

It is also important to mention that, both the framework and the query language, must be programming language independent, and more importantly, programming paradigm independent. This is important since we are looking for a generic extraction mechanism, and limiting the paradigm will in turn limit the languages and code metrics that can be accepted. Each paradigm accounts for several different code entities that can be measured, for object oriented programming for example, classes and methods related metrics can be extracted, and for aspect oriented programming, aspects, pointcuts and concerns related metrics can be extracted. Moreover, current systems can be written in any combination or programming paradigms.

The application built upon the methods proposed in this paper, must provide all the necessary elements for the intended user interaction in the source code metrics extraction process.

## 6. Conclusions

Automated tools for metric extraction are very important and a necessity for the software measurement process, without these tools, the metrics extraction process could become an almost impossible task. Even though these tools play a major role, they present a set of issues that cannot be ignored, especially the fact that are limited in the set of metrics and programming languages they recognize. This is a major issue since a tool can be of no use if the language or metric the user wants to measure is not supported. The user could use more than one tool in order to achieve his measurements goals, but it is not an efficient or convenient approach, besides, these tools are not free of cost, making the use of several tools a difficult task.

Next to those issues, new programming paradigms are gaining importance in current research, such as the aspect oriented and feature oriented paradigms, and research on

other paradigms, such as the procedural paradigm, has stalled. These changes drastically affect the validity of certain tools and the value they can provide to practitioners and researchers. More importantly, each paradigm accounts for different source code attributes, entities, and relations between them that must be measured in order to achieve the desired software quality.

In this paper, we propose a user oriented metrics tool in which the user is allowed to define metrics and incorporate programming languages to the tool according to his necessities, effectively eliminating the limitations the current software metrics tools present. It is not an easy task and depends heavily on the framework or methodology that could provide the needed mechanisms in order to achieve a generic metric extraction. By providing such a tool, the complexity of the source code measurement can be greatly reduced as well as the limits imposed by current tools.

## 7. References

- [1] IEEE Std 610.12-1990, IEEE standard glossary of software engineering terminology. <https://standards.ieee.org/findstds/standard/610.12-1990.html>
- [2] M. Scotto, A. Sillitti, G. Succi, and T. Vernazza, "A non-invasive approach to product metrics collection," *J. Syst. Archit.*, vol. 52, pp. 668–675, 2006.
- [3] T. J. McCabe, "A Complexity Measure," *IEEE Trans. Softw. Eng.*, no. 4, pp. 308–320, 1976.
- [4] M. H. Halstead, *Elements of Software Science*. New York, NY, USA: Elsevier Science Inc., 1997.
- [5] S. Chidamber and C. F. Kemerer, "A metrics suite for object oriented design," *IEEE Trans. Softw. Eng.*, vol. 20, no. 6, pp. 476–493, 1994.
- [6] F. e Abreu, "Design Metrics for Object-Oriented Software Systems," *Work. Quant. Methods Object-Oriented Syst. Dev.*, no. August, 2005.
- [7] L. Etzkorn and H. Delugach, "Towards a semantic metrics suite for object-oriented design," *Proceedings. 34th Int. Conf. Technol. Object-Oriented Lang. Syst. - TOOLS 34*, pp. 71–80, 2000.
- [8] D. Spinellis, "Tool Writing: A Forgotten Art?," *IEEE Softw.*, vol. 22, no. 4, pp. 9–11, 2005.
- [9] S. Husein and A. Oxley, "A Coupling and Cohesion Metrics Suite for Object-Oriented Software," *2009 Int. Conf. Comput. Technol. Dev.*, pp. 421–425, 2009.
- [10] J. Alghamdi, R. a Rufai, and S. M. Khan, "OOMeter: A Software Quality Assurance Tool," *Ninth Eur. Conf. Softw. Maint. Reengineering*, pp. 190–191, 2005.
- [11] A. Sillitti, A. Janes, G. Succi, and T. Vernazza, "Collecting, integrating and analyzing software metrics and personal software process data," *Conf. Proc. EUROMICRO*, pp. 336–342, 2003.

- [12] J. Bansiya and C. G. Davis, "A hierarchical model for object-oriented design quality assessment," *IEEE Trans. Softw. Eng.*, vol. 28, no. 1, pp. 4–17, 2002.
- [13] F. Fioravanti and P. Nesi, "Method and tool for assessing object-oriented projects and metrics management," *J. Syst. Softw.*, vol. 53, no. 2, pp. 111–136, 2000.
- [14] A. Sillitti, G. Succi, and S. De Panfilis, "Managing non-invasive measurement tools," *J. Syst. Archit.*, vol. 52, no. 11, pp. 676–683, 2006.
- [15] I. D. Coman, A. Sillitti, and G. Succi, "A case-study on using an Automated In-process Software Engineering Measurement and Analysis system in an industrial environment," 2009 IEEE 31st Int. Conf. Softw. Eng., pp. 89–99, 2009.
- [16] P. Emanuelsson and U. Nilsson, "A Comparative Study of Industrial Static Analysis Tools," *Electron. Notes Theor. Comput. Sci.*, vol. 217, no. C, pp. 5–21, 2008.
- [17] A. L. Baroni and F. B. e Abreu, "A Formal Library for Aiding Metrics Extraction," *Int. Work. Object-Oriented Re-Engineering*, 2003.
- [18] A. Núñez-Varela, H. G. Perez-Gonzalez, J. C. Cuevas-Tello, and C. Soubervielle-Montalvo, "A Methodology for Obtaining Universal Software Code Metrics," *Procedia Technol.*, vol. 7, no. 0, pp. 336–343, 2013.
- [19] B. Cogan, "A Generalized Structural Model of Structured Programs for Software Metrics Definition," *Softw. Qual. J.*, pp. 149–167, 2002.
- [20] C. Marinescu, R. Marinescu, and T. Girba, "Towards a simplified implementation of object-oriented design metrics," 11th IEEE Int. Softw. Metrics Symp. METRICS05, no. Section 6, p. 11 TS – CrossRef, 2005.
- [21] E. H. Alikacem and H. A. Sahraoui, "Generic Metric Extraction Framework," in *IWSM/ MetriKon 2006*, 2006.
- [22] E. H. Alikacem and H. a. Sahraoui, "A Metric Extraction Framework Based on a High-Level Description Language," 2009 Ninth IEEE Int. Work. Conf. Source Code Anal. Manip., pp. 159–167, 2009.
- [23] S. Allier, S. Vaucher, B. Dufour, and H. Sahraoui, "Deriving coupling metrics from call graphs," *Proc. - 10th IEEE Int. Work. Conf. Source Code Anal. Manip. SCAM 2010*, pp. 43–52, 2010.
- [24] Unified Modeling Language. <http://www.uml.org/>
- [25] Object Constraint Language - Object Management Group. <http://www.omg.org/spec/OCL>
- [26] Z. Budimac, G. Rakic, M. Hericko, and C. Gerlec, "Towards the Better Software Metrics Tool," 2012 16th Eur. Conf. Softw. Maint. Reengineering, pp. 491–494, 2012.
- [27] D. P. Darcy and C. F. Kemerer, "OO Metrics in Practice," *IEEE Softw.*, vol. 22, no. 6, pp. 17–19, 2005.
- [28] R. Lincke, J. Lundberg, and W. Löwe, "Comparing software metrics tools," *Proc. 2008 Int. Symp. Softw. Test. Anal. - ISSTA '08*, p. 131, 2008.



- [29] M. Scotto, A. Sillitti, G. Succi, and T. Vernazza, "A relational approach to software metrics," Proc. 2004 ACM Symp. Appl. Comput. SAC 04, p. 1536, 2004.
- [30] Núñez-Varela, A. S., Pérez-González, H. G., Martínez-Pérez, F. E., & Cuevas-Tello, J. (2016). Building a user oriented application for generic source code metrics extraction from a metrics framework. In 2016 4th International Conference in Software Engineering Research and Innovation. <http://doi.org/10.1109/CONISOFT.2016.13>

# Chapter # 12

## PMBOK and Essence: Partners for IoT Projects

**Marcel J. Simonette**  
Escola Politécnica  
Universidade de São  
Paulo São Paulo, Brazil  
marceljs@usp.br

**Mário E. S. Magalhães**  
CEST  
Universidade de São  
Paulo  
São Paulo, Brazil  
m.e.magalhaes@ieee.org

**Edison Spina**  
Escola Politécnica  
Universidade de São  
Paulo  
São Paulo, Brazil  
spina@usp.br

### 1. Introduction

Internet of Things (IoT) is a concept that has several components, such as technology, human factors, business process, and engineering standards [1, 2, 3, and 4]. Model based system engineering build models that allows the identification of the engineering activities that are present in system life cycle; furthermore, engineers use systems models to understand problems, develop candidate solutions, and validate their decisions [5].

Systems models allow the identification of risks and obstacles to solutions implementation. They represent both the desire outcome of the design process and what the system will look like [6].

To be successful, companies that takes part of the IoT ecosystem must be able to respond quickly to the changing demands of both stakeholders and technology, and must have appropriate levels of engineering discipline [7]. Moreover, there is an intrinsic complexity in IoT systems, as there are very simple software running on basic sensors and other simple devices, and, at the same time, there are the dependability issues of the components needed to process, analyze and respond to the vast amounts of data that are produced.

IoT systems development need to use many methods and practices to deal with its complexity, however, these methods and practices need a common ground ba-

sed on project management body of knowledge and the systems and software engineering. The successful implementation of IoT systems requires not only engineers' technical skills but managerial traits as well. The combination of technical skills and management principles allow systems engineers address both the technical and managerial issues that are present in IoT systems life cycle [8, 9, 10].

The authors' research concerns the relationship between PMBOK process and Essence Kernel activity spaces, which allows the consideration of the project management body of knowledge in IoT development endeavors [11]. In addition, the author use the BRISCS Mosaic Model as a systems model to identify barriers that must be consider in IoT systems and software.

## 2. Brics mosaic model

To model IoT systems it is necessary to develop a context-aware socio-technical model, which allows the representation of the knowledge diversity present in system context and stakeholders objectives [12, 13, 14, and 15].

Based both on concept plans of Next Generation Networks (NGN) [16], and on the results and experience of two projects of the 7th Framework Programme for Research and Technological Development (FP7): CSA for Global RFID-related Activities and Standardization (CASAGRAS2) [17], and Internet of Things Architecture (IoT-A) [18], the BRICS Mosaic Model represents the engineering and non-engineering aspects required for IoT systems. The Model provides a context for a representative concept that organizes the characteristics of the IoT universe into planes of a cylindrical mosaic, which are represented in Figure 1. Each plane of the cylindrical model of the Mosaic allows the identification of a research area for IoT. That is the reason why it is named as BRICS, an acronym to: Building blocks of Research for the Internet-Connected objects. BRICS Mosaic planes represent solution views, and are considered "planes of functionality". The first plane (Figure 1) represents the Technological view, and the other planes are representation of: Security, Business Process, Integrated Management and Control, Regulations, and Human Factors. If any other view is identified in an IoT solution, it may be another plane in Mosaic.

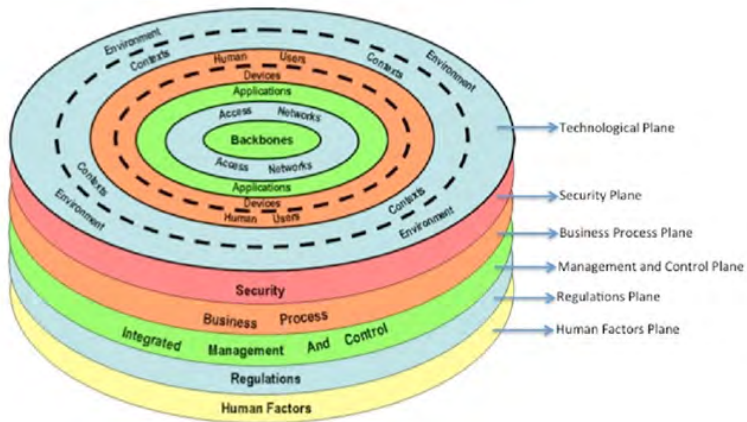


Figure 1. BRICS MOSAIC Model

Each plane has the same set of dimensions that drive, influence and affect the development of IoT systems (Figure 2). No single plane, and no single dimension of a plane, can yield a satisfactory model for an IoT system.

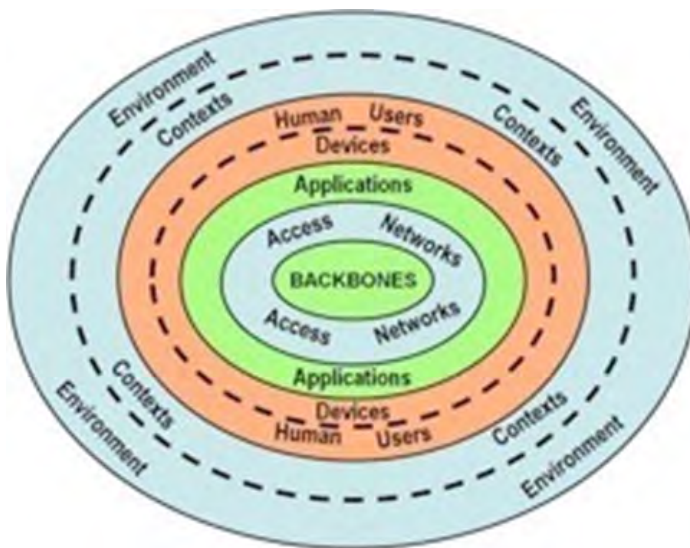


Figure 2: Feasibility Barrier Factors

The set of dimensions are the Feasibility Factors (FF). Each FF is a barrier, a restriction to be overcome. All the planes of functionality in BRICS Mosaic Model have the same set of FF, and each Factor in each plane is represented as the zone between two concentric circles in this plane. Furthermore, each zone represents a different medium in which information is carried over. The concentric circles in each plane represent the fact that data can transit from any point to any other point. Furthermore, there are different software systems in each of these dimensions. Reference [19] presents a full example of the use of BRICS Mosaic Model and FF.

### 3. Essence

Although there are several methods and practices to conduct the software development process, there is a set of essential elements and activities that are universal and present in all software systems projects. These elements and activities deal with “things we always work with” (Figure 2) and “things we always do” (Figure 3) in a software system project [20]. These elements provide an integrated set of universal elements that are present in software systems development projects, enabling to measure progress and health of software projects. Furthermore, the set of essential elements facilitates the communication and the identification of the actions to be taken throughout a project lifecycle [20, 21].

Essence Kernel was first published in the Software Engineering Method and Theory (SEMAT) submission to respond to the Object Management Group (OMG) call: “Foundation for Agile Creation and Enactment of Software Engineering” [22]; actually, Essence Kernel and Language is an OMG standard [23]. More than a conceptual model, the kernel provides:

A framework for project teams to reflect on the progresses of an endeavor and the state of their efforts.

A common ground to discuss, to improve, to compare, and to share the Software Engineering best practices and methods.

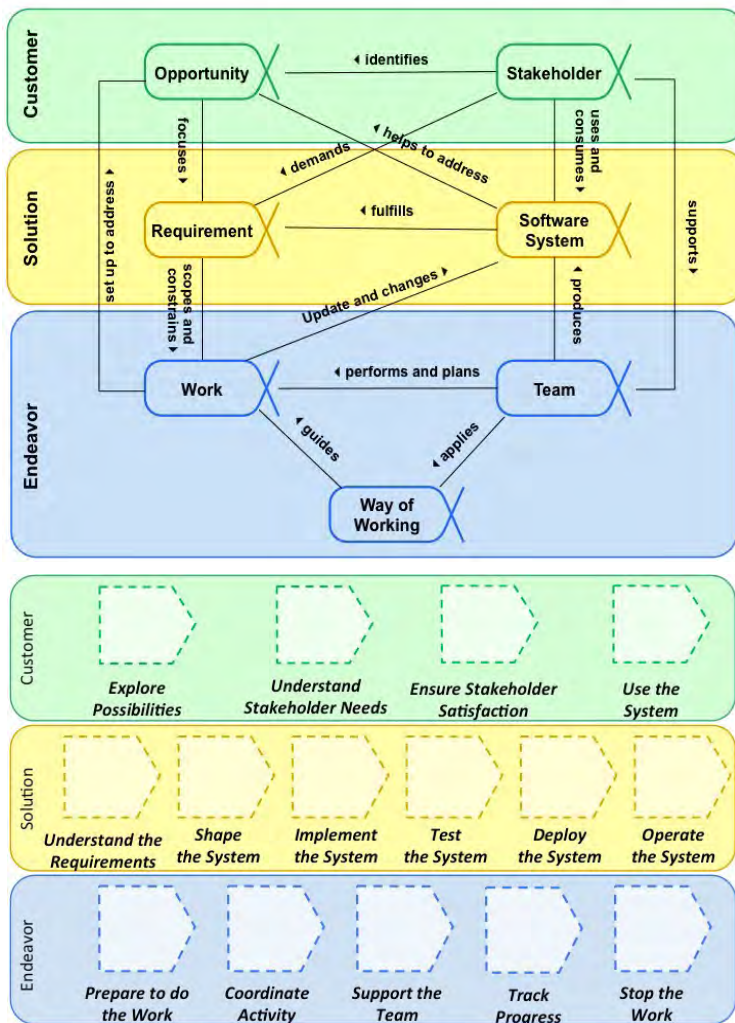


Figure 2. "Things we always do"

- A framework for project teams to assemble practices to assemble practices from different origins to continuously improve the way of working.
- A framework for defining metrics that are improve the independent of practices, to evaluate both of the development software and the methods used to develop it.

- A way to help project teams to understand (i) their present scenario, (ii) what they should do to advance in the project process, and (iii) which practices of the process they need to improve.

Essence Kernel is organized into three specific dimensions of software development: Customer, Solution, and Endeavor. They are the areas of concern of the standard [21]. Each of these areas of concern has a set of activity spaces, which are a representation of essential things that have to be done to develop software systems. Activity Spaces aims to establish the way to a successful software engineering endeavor. Figure 2 represents these activity spaces for the three areas of concern.

## 4. PMBOK process groups

The PMBOK - Project Management Body of Knowledge - represents the results of a global effort to support and to recognize the application of knowledge, processes, skills, tools, and techniques to have a significant impact on project success. It recognizes ten areas of project knowledge management, the double nature of project process and five groups of project management processes.

The project knowledge areas are [24]: (i) Project Integration Management, (ii) Project Scope Management, (iii) Project Time Management, (iv) Project Cost Management, (v) Project Quality Management, (vi) Project Human Resource Management, (vii) Project Communications Management, (viii) Project Risk Management, (ix) Project Procurement Management and (x) Project Stakeholder Management.

The double nature of project processes regards (i) Project management processes and (ii) Product-oriented processes. Even though the PMBOK Guide recognizes the interaction and overlap between these two groups, it describes only project management processes.






The project management processes are described by their purposes, interactions, and integrations. They are grouped into five categories, known as Project Management Process Group, or simply as Process Group [24]:

- Initiating Process Group.

- Planning Process Group.
- Executing Process Group.
- Monitoring and controlling Process Group.
- Closing Process Group.

## 5. Essence and PMBOK

A comparison context considering the Activity Spaces of Essence Standard and the PMBOK Process Groups and Knowledge Areas has been presented by the authors in [11]. Tables 1, 2, and 3, from [11], describe the PMBOK Management Processes present in each Essence Activity Spaces. The five PMBOK Process Groups are represented by a different color pattern:

-  ! Initiating Process Group
- 
- 
-  ! Planning Process Group
- 

! Monitoring and Controlling Process Group

! Executing Process Group

! Closing Process Group

Table 1 indicates that in the Customer area of concern, the project chart is developed, the risks and the stakeholders are identified, the project scope is validated and the project is closed.

Table 2 presents the PMBOK Management Processes that are present in Solution Activity Spaces.

As expected, most PMBOK Management Processes

(74%) are present in the Endeavor Activity Spaces (Table 3). In the Endeavor area of concern, it is necessary to effectively plan, lead, and monitor the efforts of the team. The activity spaces in this area: cover the formation and support of the



team implementing the software system, in addition to planning and the coordinating of the work.

## 6. IOT Projects

IoT projects have a complexity that demands new domain-specific practices. Jacobson et al. [7] suggest the need of practices that handle with dependability issues, such as:

- Practices to deal with reliability and availability of distributed software systems.
- Practices to develop mobile systems that must deal with availability, security and safety of software systems.
- Practices to deal with the whole idea of IoT, which is to sense-analyze-activate without a human in the loop.

In spite of the fact that it is necessary a new set of practices to deal with IoT issues, the authors agree with Jacobson et al.[7] that new management practices are not necessary, especially to the Industrial IoT, which usually makes use of PMBOK to manage its projects.

## 7. Conclusion and future work

No method is perfect, and finding perfect partners is almost impossible. However, it is possible for different methods to complement each other, and the mapping of the PMBOK Process Groups in the Essence Activity Spaces represents this complementary relationship.

The mapping represented in Tables 1, 2, and 3 allows project managers and the software development team to go beyond the inputs and outputs of each Activity Space in terms of the alphas states of Essence Kernel. It allows identifying which Project Management processes should be considered in each Activity Space.

Table 1. Customer activity space, pmbok management process and knowledge areas mapping

	understand the requirements	shape the system	implement the system	test the system	deploy the system	operate the system
4. Project Integration Management						
5. Project Scope Management	5.2. Collect Requirements 5.3. Define Scope					
6. Project Time Management		6.2. Define Activities				
7. Project Cost Management						
8. Project Quality Management				8.2. Perform Quality Assurance		
9. Project Human Resources Management						
10. Project Communication Management						
11. Project Risk Management	11.3. Perform Qualitative Risk Analysis 11.4. Perform Quantitative Risk Analysis					
12. Project Procurement Management			12.2. Conduct Procurements			
13. Project Stakeholder Management						

Table 2: Solution activity space, pmbok management process and knowledge areas mapping

	Explore possibilities	Involve the stakeholders	Ensure stakeholders satisfaction	Use the system
4. Project Integration Management	4.1. Develop Project Chart			4.6. Close Project Phase
5. Project Scope Management			5.5. Validate Scope	
6. Project Time Management				
7. Project Cost Management				
8. Project Quality Management				
9. Project Human Resources Management				
10. Project Communication Management				
11. Project Risk Management	11.2. Identify Risks			
12. Project Procurement Management				
13. Project Stakeholder Management		13.1. Identify Stakeholders		

Table 3. Endeavor activity space, pmbok management process and knowledge areas mapping

	Prepare to do the work	Coordinate activity	Support the team	Track the progress	Stop the work
<b>4. Project Integration Management</b>	4.2. Develop Project Management Plan	4.5. Perform Integrated Changes Control	4.3. Direct and Manage Project Work	4.4. Monitor and Control Project Work	4.6. Close Project Phase
<b>5. Project Scope Management</b>	5.1. Plan Scope Management 5.4. Create WBS			5.6. Control Scope	
<b>6. Project Time Management</b>	6.1. Plan Schedule Management 6.4. Estimate Activity Resources 6.5 Estimate Activity Duration	6.3. Sequence Activities 6.6. Develop Schedule		6.7 Control Schedule	
<b>7. Project Cost Management</b>	7.1. Plan Cost Management 7.2. Estimate Cost 7.3. Determine Budget			7.4. Control Costs	
<b>8. Project Quality Management</b>	8.1. Plan Quality Management			8.3. Control Quality	
<b>9. Project Human Resources Management</b>	9.1. Plan Human Resource Management	9.2. Acquire Project Team 9.4. Manage Project Team	9.3. Develop Project Team		
<b>10. Project Communication Management</b>	10.1. Plan Communications Management	10.2 Manage Communications		10.3. Control Communications	
<b>11. Project Risk Management</b>	11.1. Plan Risk Management 11.5. Plan Risk Responses				
<b>12. Project Procurement Management</b>	12.1. Plan Procurement Management			12.3. Control Procurements	12.4. Close Procurements
<b>13. Project Stakeholder Management</b>	13.2 Plan Stakeholder Management	13.3 Manage Stakeholder Engagement		13.4. Control Stakeholder Engagement	

The BRICS Mosaic Model, and its FF, allows the identification of the barriers, the restriction to be overcome in each technological view of a IoT systems. Identifying barriers in IoT projects is the first step to select the set practices to overcome it. However, these practices must not be combined in different isolated methods, without a widely accepted common ground [7].

To deal with the software development practices necessary to overcome the barriers in IoT projects, Essence kernel, part of the OMG standard Essence [23], can provide a foundation, the common ground, that allows the assembly of a comprehensive practice library containing the practices needed for the IoT systems domain [7].

Essence Kernel also allows the definition of lifecycle apart from the set of practices used by the development team. It is an important feature in a complex domain as IoT projects, which demands a kind of governance that needs to capture checkpoints and manage several life cycles, not constraining them to any pre-defined type of style of software development process.

IoT systems project success depends on management. In other words, IoT projects may fail because management may assume that these projects are “just another project”. IoT systems complexity demands management and engineering discipline. BRICS Mosaic Model allows the identification of the barriers that must be overcome. Essence kernel provides the common ground to the set of practices to deal with these barriers. The identified relations among PMBOK Project Management Knowledge Areas, Processes Groups, and Essence Activity Spaces allows the management of the complexity in software engineering IoT projects endeavors.

## 8. Acknowledgment

This work was partly supported by the Society and Technology Study Center (or, CEST – Centro de Estudos Sociedade e Tecnologia, in Portuguese) at Universidade de São Paulo.

## 9. References

- [1] Gerd Kortuem, Arosha K. Bandara, Neil Smith, Mike Richards, Marian Petre, “Educating the Internet-of-Things Generation,” *Computer*, vol. 46, no. 2, pp. 53-61, Feb., 2013
- [2] Mohamed Ali Feki, Fahim Kawsar, Mathieu Boussard, Lieven Trappeniers, “The Internet of Things: The Next Technological Revolution,” *Computer*, vol. 46, no. 2, pp. 24-25, Feb., 2013
- [3] D. Roggen, Gerhard Troster, P. Lukowicz, A. Ferscha, Jose del R. Millan, R. Chavarriaga, “Opportunistic human activity and context recognition,” *Computer*, vol. 46, no. 2, pp. 36-45, Feb., 2013
- [4] G. Schwartz, E. Spina, J.R.A. Amazonas, “Internet of the Future NonEngineering Challenges”. In: *The 3rd International MultiConference on Engineering and Technological Innovation Proceedings*. Winter Garden, FL: IIIS International Institute of Informatics and Systems, 2010. v. 1. p. 146-151.
- [5] Loyd Baker, Paul Clemente, Bob Cohen, Larry Permenter, Byron Purves, and Pete Salmon, *Foundational Concepts for Model Driven System Design*. INCOSE International Symposium, Vol.6, No.1, Jul. 1996, pp.1179-1185. DOI:10.1002/j.2334-5837.1996.tb02139.x

- [6] Chris Piaszczyk, Model Based Systems Engineering with the Department of Defense Architectural Framework, *Systems Engineering*, vol.: 14, no.: 3, pp.: 305-326. DOI: 10.1002/sys.20180
- [7] I. Jacobson, I. Spence, P. W. Ng. Is There a Single Method for the Internet of Things? Ivar Jacobson International white paper. Available at: <https://www.ivarjacobson.com/publications/white-papers/industrial-internet-needs-many-methods>
- [8] Alexander Kossiakoff, William N. Sweet, Samuel J. Seymour, Steven M. Biemer, *Systems Engineering Principles and Practice*, 2nd Edition, John Wiley & Sons, Inc., Hoboken, New Jersey, 2011.
- [9] Amira Sharon, Olivier L. de Weck, and Dov Dori. Project management vs. systems engineering management: A practitioners' view on integrating the project and product domains. *Systems Engineering*, Vol.14, No.4, 2011, pp. 427-440. DOI=10.1002/sys.20187
- [10] Andrew P. Sage, *Systems engineering: Fundamental limits and future prospects*. Proceedings of the IEEE, vol.69, no.2, pp.158-166, Feb. 1981. DOI=10.1109/PROC.1981.11948
- [11] M. J. Simonette, M. E. S. Magalhães and E. Spina, "PMBOK Five Process Groups and Essence Standard: Perfect Partners?" 2016 4th International Conference in Software Engineering Research and Innovation (CONISOFT), Puebla, 2016, pp. 53-58. DOI: 10.1109/CONISOFT.2016.17
- [12] Anind K. Dey, Gregory D. Abowd, and Daniel Salber, a conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, Vol.16, No. 2, Dec. 2001, pp. 97 - 166. DOI =10.1207/S15327051HCI16234\_02
- [13] Gerhard Fischer, *User Modeling in Human-Computer Interaction. User Modeling and User-Adapted Interaction*, Vol.11, No.1-2, Mar. 2001, pp.65-86. DOI=10.1023/A:1011145532042
- [14] Stefan Carmien, Melissa Dawe, Gerhard Fischer, Andrew Gorman, Anja Kintsch, and James F. Sullivan, Socio-technical environments supporting people with cognitive disabilities using public transportation. *ACM Transaction on Computer-Human Interaction*, Vol. 12, No. 2, Jun. 2005, pp. 233-262. DOI=10.1145/1067860.1067865
- [15] Zoran Bojkovic, Bojan Bakmaz, Miograd Bakmaz, Some Challenging Issues for Internet of Things Realization. Proc. 12th International Conference on Data Networks, Communications, Computers (DNCOCO '13), Lemesos, Cyprus, Mar. 2013, pp. 63-70. WSEAS Press, ISBN:978-1-61804-169-2, ISSN: 1790-51117. Available at: <http://www.wseas.us/e-library/conferences/2013/Lemesos/TELSYS/TELSYS-08.pdf>
- [16] ITU-T Y.2011, Next Generation Networks - Frameworks and functional architecture models, General principles and general reference model for Next Generation Networks, 10/2004.
- [17] CASAGRAS2 - CSA for Global RFID-related Activities and Standardization. Project website available at: ([http:// cordis.europa.eu/projects/rcn/85786\\_en.html](http://cordis.europa.eu/projects/rcn/85786_en.html)).

- [18] IoT-A – Internet of Things Architecture. Project website available at: ([http://cordis.europa.eu/projects/rcn/95713\\_en.html](http://cordis.europa.eu/projects/rcn/95713_en.html))
- [19] M. J. Simonette, R. Filev, D. Gabos, J. R. Amazonas, E. Spina. BRICS Mosaic Model for IoT Feasibility Barriers. In: Recent Researches in Electrical Engineering - Proceedings of the 13th International Conference on Circuits, Systems, Electronics, Control & Signal Processing (CSECS '14), Lisbon, Portugal, October 30 – November 1, 2014. Ed: George Vachtsevanos, Cornelia Aida Bulucea, Nikos E. Mastorakis, Klimis Ntalianis, World Scientific and Engineering Academy and Society (WSEAS), 2013. p. 180-188. ISBN 978-960-474-392-6. ISSN: 1790-5117. Available at: <http://www.wseas.us/e-library/conferences/2014/Lisbon/ELEL/ELEL-21.pdf>
- [20] I. Jacobson, P. W. Ng, P. E. McMahon, I. Spence, and S. Lidman, “The essence of software engineering: The SEMAT Kernel,” in Queue, vol. 10, no. 10, ACM, Oct. 2012. Doi: 10.1145/2381996.2389616
- [21] I. Jacobson, P. W. Ng, P. E. McMahon, I. Spence, and S. Lidman, The Essence of Software Engineering: Applying the SEMAT Kernel. Addison-Wesley Professional, New Jersey, 2013.
- [22] OMG - Foundation for Agile Creation and Enactment of Software Engineering RFP. [Online]. Available at: <http://www.omg.org/cgi-bin/doc?ad/2011-6-26>
- [23] OMG - Essence Kernel and Language for Software Engineering Methods. [Online]. Available at: <http://www.omg.org/spec/Essence/1.0/>
- [24] PMBOK - Project Management Institute, A Guide to the Project Management Body of Knowledge (PMBOK® Guide) --Fifth Ed. Project Management Institute, 2013.



# PART 2 MODELING

---



# Chapter # 13

## Automotive Safety Requirements Specification

Jorge Aguilar Cisneros, Emmanuel  
Gulias Mata, Luis Fernando  
Torreblanca Macías.  
Departamento de Ingenierías,  
Universidad Popular Autónoma del  
Estado de Puebla  
21 sur #1103 Col. Santiago, Puebla Pue.,  
México C.P. 72310  
jorge.aguilar@upaep.edu.mx,  
{emmanuel.gulias01, luisfernando.  
torreblanca}@upaep.mx

Carlos Fernández y Fernández  
Instituto de Computación, Universidad  
Tecnológica de la Mixteca, carretera a  
Acatlima Km. 2.5 Huajuapán de León,  
Oax., México C.P. 69000,  
caff@mixteco.utm.mx

### 1. Introduction

Modern vehicles are definitely “software-intensive” systems and software is now controlling an increasing number of traditional functions [14]. The need for more efficient control has motivated the introduction of a considerable amount of software. For example, the hybrid model Ampera from Opel, manufactured by General Motors, has over 10 million lines of code [15] and a modern high-end car, which features around 100 million lines of code executing on ECUs [16], and this number is planned to grow to 200-300 million in the near future.

Unfortunately, together with this scenario, the occurrence of multiple automotive recalls has been increasing [6]. For this reason, the automotive engineers should use automotive standards like ISO 26262 and spend more time in the analysis and design phases. In particular, in this paper, we tackled the software safety requirements specification in order to diminish the number of vehicle recalls and increase the automotive software security.

As defined in the IEEE Standard Glossary of Software Engineering Terminology [1], a requirement is a condition or capability needed by a user to solve a problem or achieve



an objective. In other words, requirements define how a system will behave. In software development, requirements specification is a crucial step in the development process, as more often than not, it dictates the overall outcome of the system. Defective requirements will usually send the development process spiraling down towards failure. [2].

Requirements engineering is the software engineering branch that involves accurate description of the objectives and methods to be used in a software project. Not only does it cover requirements specification, but also their management throughout the life cycle. This practice manages requirements from the moment they are defined, passing through any changes made, to the moment they are met. It is one of the more important aspects of software development.

However, are they really that important? Is it really worth it investing so many resources in tools and training? A thorough requirements definition involves a great deal of effort and resources. However, it is only necessary to analyze certain real life examples of projects with defective requirements.

On September 14, 2005, an Air Route Traffic Control Center in Los Angeles, CA. lost all communication with 400 airplanes. [3] The reason? A glitch in the system which made it run for 50 days before it crashed, needing a manual reboot before that happened. Thanks to the collision avoidance systems built in the aircraft, a situation that could have ended in tragedy was averted. The troubling aspect of this situation is that developers were aware of the error. However, they didn't inquire in the repercussions it could have when fully implemented.

On June 4, 1996, the European Space Agency launched Ariane 5, a massive rocket whose mission was to launch three satellites into orbit. After 10 years of development and \$7 billion to produce it, most people were eager to see it in action. After 37 seconds, the rocket exploded [4]. It turns out, the system tried to store a 64-bit floating point number into a 16-bit integer variable. When it obviously couldn't, the system shut down and the disaster occurred. The worst part of the matter, is that the defective piece of code wasn't even necessary in this new Ariane model. However, a faulty requirement analysis didn't take that into account and allowed it to stay.

Simple errors. Simple solutions. Enormous consequences. In both cases, the errors triggered a total shutdown of operations. The systems involved are often known as mission critical systems. One of them could even be just a part of a bigger and more complex system, which will then affect the whole thing. In an online banking system,

this could mean thousands of transactions to be lost, and angry clients making calls and contacting tech support. However, what if the failure endangered a person's life? Let us say, a commercial jet's engine, or a car's braking system. On the other hand, there are also non-mission critical systems, which in the event of failure would only be considered a minor nuisance to be fixed when possible. While their malfunction might not be as catastrophic as mission critical systems, that does not mean they should be allowed to have sloppy requirements.

This article's interest lies on non-mission critical software systems for the automotive industry, specifically in the aspect of vehicular safety. Donald Firesmith, of the Software Engineering Institute, stated that there are still many defects found in requirements specification [5]. Recently, software innovation focus for the automotive sector has shifted towards improvement of electronic control units (ECUs). But at the same time, the number of defective automobiles delivered to customers has increased accordingly [6]. This is probably due to the lack of a unified design schema for automotive software.

Even though there is a considerable quantity of methods and techniques used in requirements specification, identification, analysis, management and verification, it is unclear why the produced systems lack the desired quality. To address this issue, we propose a requirement modeling style as indicated by ISO 26262 functional safety standard for road vehicles, and following SysML, a modeling language specification for systems engineering applications.

## 2. Background

ISO 26262 is to be used for mass-produced passenger cars of up to 3500 kg of weight that have at least one electrical or electronic (E/E) systems installed. 26262's aim is to address possible hazards that could impact the behavior or functionality of E/E safety systems. Part 6 of this standard, which is the main source of information to be used, specifies the requirements for product development at the software level. It includes initial requirements specification, architectural and unit design, testing, software integration and verification. We will be focusing on the safety requirements specification section as we will explain how to represent these automotive requirements in order to reach the security required for automotive systems development.

Safety requirements have to be specified and detailed in a hierarchical structure. The structure and dependencies of safety requirements used in ISO 26262 are shown in Figure 1.

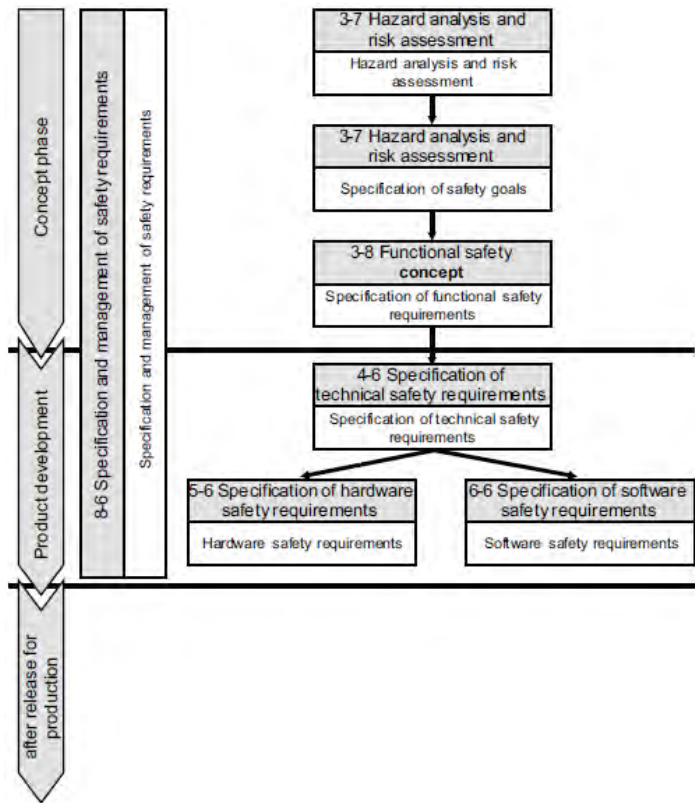


Figure 1. Structure of safety requirements

The automotive safety requirements specification has to be correct with respect to their attributes and characteristics. In accordance with ISO 26262, the safety requirements characteristics are:

1. Unambiguous and comprehensible. That mean, the requirement is unambiguous if it is a common understanding of the meaning of the requirement. The requirement is comprehensible if the reader at an adjacent abstraction level understands its meaning.
2. Atomic. It is atomic when it is formulated in such a way that it cannot be divided into more than one safety requirement at the considered level.
3. Internally consistent. Each safety requirement contains no contradictions with itself.
4. External Consistency. Multiple safety requirements do not contradict each other.

- 5. Feasible and verifiable. It can be implemented with the constraints of the item development.

The safety requirements attributes are:

- 1. A unique identification remaining unchanged throughout the safety lifecycle.
- 2. A status
- 3. An ASIL

Safety requirements constitute all requirements aimed at achieving and ensuring the required ASIL's.

Automotive Safety Integration Level (ASIL) is a risk classification scheme defined in ISO 26262 [18] that indicates the level of hazard expected in a given scenario. This is done by analyzing the risk's potential severity, the exposure it carries and how controllable it is. ISO 26262 explains 4 levels of ASIL: A, B, C, D where level A is the lowest level of risk and D is the highest level. It is implied that a system credited with ASIL D also complies with the lower levels of the standard (see Figure 2).

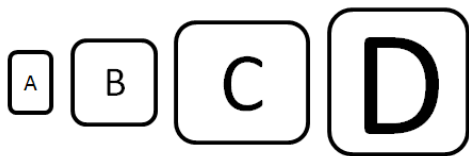


Figure 2. ISO 26262, ASILs Level.

There are four safety levels defined at ISO26262 Standard: A, B, C and D. Additionally, there are three criteria to determine someone of these levels: severity, probability of exposure and controllability.

- Severity. Quantitative measurement of the consequences of a car accident. (See Table 1).

Table 1. Classes of Severity [18]

Severity			
S0	S1	S2	S3
No injury	light and moderate injuries	severe and life-threatening injuries (probable survival)	life-threatening injuries (uncertain survival), fatal injuries

- **Probability of Exposure.** Qualitative measurement of the possibility of the system (and the user) being in a situation where the occurrence of the accident is conceivable (See Table 2).

**Table 2. Classes of Probability of Exposure [18]**

Probability of Exposure				
E0	E1	E2	E3	E4
improbable	very low probability	low probability	medium probability	high probability

- **Controllability.** Qualitative measurement of the capability of the user to avoid a dangerous situation. This criterion is specific to the automotive domain where the user (the driver) can exercise a certain control on a permissive system (the vehicle does not inhibit unforeseen behaviors). (See Table 3).

**Table 3. Classes of Controllability [18]**

Controllability			
C0	C1	C2	C3
controllable in general	simply controllable	normally controllable	difficult to control or uncontrollable

These three criteria allow determining in a systematic way the ASIL of a system or of one of its features. (See Table 4).

**Table 4. Automotive Safety Integrity Levels (ISO 2008)**

Severity	Exposure	Controllability		
		C1	C2	C3
S1	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	A
	E4	QM	A	B
S2	E1	QM	QM	QM
	E2	QM	QM	A
	E3	QM	A	B
	E4	A	B	C
S3	E1	QM	QM	A
	E2	QM	A	B
	E3	A	B	C
	E4	B	C	D

If the evaluation leads to an ASIL quotation lower than level A, a QM quotation (for Quality Management) is assigned to the event and no safety requirements are defined for the system. A QM quotation means that a Quality Management Process is mandatory and sufficient to meet the safety goal [17].

Let us say we follow the guidelines established by ISO 26262 and write requirements that comply with ASIL D.

Is it even necessary to model those requirements? 75% of automotive companies design by using iterating methodologies such as Rapid Control Prototyping (RCP) [7]. While these methods are not to be considered as true requirements engineering, they are quite useful for providing an ampler look of the system. Model Based Requirement Engineering and Model Based Testing reduce costs, testing times and defects injected [8]. Because of this, accurate requirement modeling helps enhance the overall definition of the process.

There are several different proposals for automotive requirement modeling. The Aspect-Oriented Requirements Engineering (AORE) methodology enables to model the collaboration among the distributed embedded automotive software systems in terms of aspects and generate multiple product lines while assuring a set of non-functional requirements including safety, performance and cost. [9].

Formal Service description Language (ForSeL) for model-based requirements engineering. The basic notion in ForSeL is a service representing a functional requirement. Each service describes a system “re”-action that is triggered by a set of input actions – (but only) if an additional precondition holds. The functional part of a specification is then obtained by the combination of a finite number of services. There are two kinds of preconditions which are often mixed up in practice: sufficient and necessary preconditions. *ForSeL describes functional requirements in terms of formal services* [10].

The purpose of the EAST-ADL language is to capture automotive electrical and electronic systems with sufficient detail to allow modeling for documentation, design, analysis, and synthesis. This language specification describes how information needed for relevant analysis and synthesis can be captured but does not define how the analysis or synthesis should be done. EAST – ADL has constructs that deal with the safety requirements specification. The requirements part will be compliant with UML2. Figure 3 shows a diagram for a safety requirement [19].

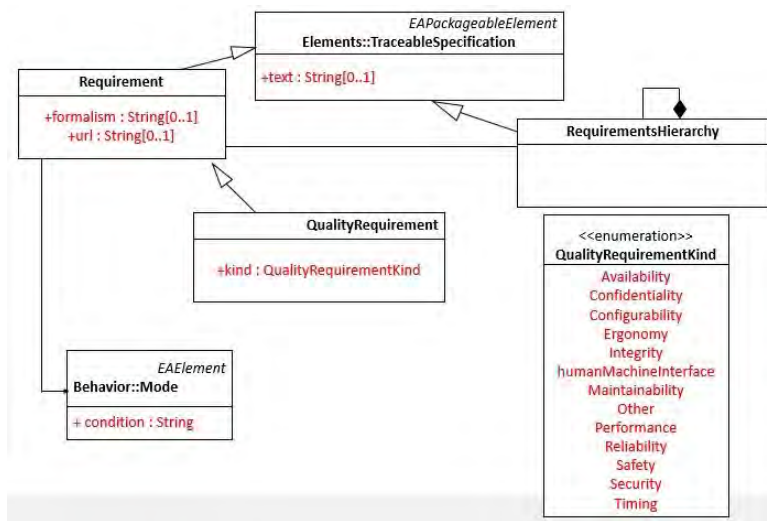


Figure 3. Diagram for requirements overview.

EAST-ADL2 is an architecture description language specifically dedicated to automotive systems. It describes the overall functioning of a system in a vehicle, analysis, design, implementation and operational level [11]. Aspectual models cover three different modeling techniques (EAST-ADL2, timed automata and signal matrix) and combines them into UPPAAL specifications for easy simulation and verification [12]. Despite the wide array of methodologies to choose from, we have decided to use just one: SysML.

The Systems Modeling Language (SysML) is a general-purpose modeling language for systems engineering applications that supports the specification, analysis, design, verification and validation of a wide array of systems or subsystems. These systems may include hardware, software, information, processes, personnel, and facilities. It was developed as an extension of the Unified Modeling Language (UML 2), which is more software oriented. It is more flexible and supports requirement engineering diagrams, which allow for performance and qualitative analyses, useful in safety requirements. SysML reuses a subset of UML 2 and provides additional extensions to satisfy the requirements of the language. It is designed to provide constructs for modeling a wide range of systems engineering problems. SysML is particularly effective in specifying requirements.

SysML provides modeling constructs to represent text-based requirements. The requirements diagram can depict the requirements in graphical, tabular, or tree structure format. (See Figures 4-7).

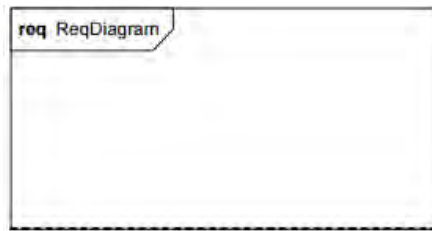


Figure 4. Requirement diagram

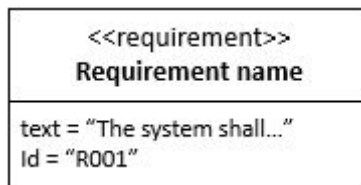


Figure 5. Requirement specification

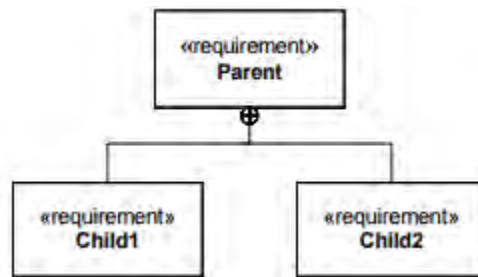


Figure 6. Requirement containment relationship

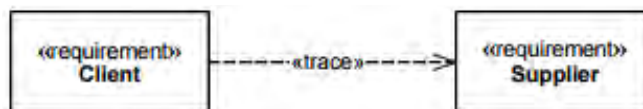


Figure 7. Trace Dependency

The diagrams above are only some of the diagram elements include in OMG SysML Version 1.4 [20]

There are different ways to requirement specification: Natural Language, informal specification, semi-formal specification and formal specification. For this paper, we consider Natural language and informal specification like informal specification.



An example of a requirement in natural language or informal specification could be: *“Signal lowFuelLevelWarning shall be set to Active when input totalFuelLevel is below a predefined level. This level shall be 10% for tank size equal to or below 900 liters and 7% for tank sizes larger than 900 liters. The tank size is determined by the parameters fuelTankSizeLeft and fuelTankSizeRight”* [21]. Writing requirements in natural language is subject to ambiguity and imprecision [22].

Structured Analysis is a semi-formal type of specification technique which combines the use of natural language and graphical symbol with semantics. It has less potential for ambiguity but requires technical Skills. For this type of requirements specification, we can use SysML.

Specification language Z is considered a formal method based on its definition as “a general description of the use of mathematical notations such as logic and set theory to describe systems specifications and software design together with techniques of validation and verification based on mathematics” [23].

### 3. Development

When writing requirements for automotive safety, it is important to comply with ASIL specifications [24]. As seen in Table 5, taken directly from ISO 26262 [13], using natural language is highly encouraged for all levels of ASIL. However, for levels A and B, informal notation is also recommended for use. Lastly, levels B through D can be written in a semi-formal manner. “++” indicates that the method is highly recommended for the identified ASIL; “+” indicates that the method is recommended for the identified ASIL; “o” indicates that the method has no recommendation for or against its usage for the identified ASIL.

Table 5 - Notations for software unit design

Methods		ASIL			
		A	B	C	D
1a	Natural language	++	++	++	++
1b	Informal notations	++	++	+	+
1c	Semi-formal notations	+	++	++	++
1d	Formal notations	+	+	+	+

Once the requirements are written and verified they comply with the desired level of ASIL, they should be graphically represented using the corresponding model described by SysML. SysML introduces requirements diagrams in comparison to UML. As shown in

Figure 8, requirement diagrams model all requirements for a system, using relationships between them to establish a client-supplier connection in which the first depends on the latter, and as such, any changes in the supply will provoke changes in the client. These relationships can be defined as:

- » A requirement that derives in another one
- » A requirement that is satisfied by a block, module or subsystem
- » A requirement that is verified by a certain test case
- » A requirement that is broken down and refined in a specified use case
- » A requirement that can be traced back to its origin

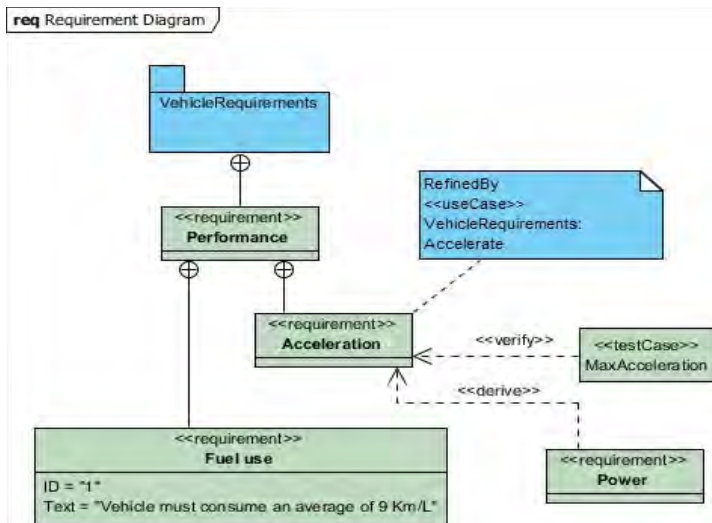


Figure 8. Requirement Diagram Example

In a SysML requirement diagram, objects with the `<<requirement>>` tag must have an ID to identify them, and the text related to the requirement. Different objects are to be considered as side notes, each with their own title (such as `RefinedBy`, `VerifiedBy`, `SatisfiedBy`) and their own tag like `<<useCase>>`, `<<testCase>>` or `<<block>>`. The relationships defined earlier in this document, are to be set in the lines connecting each object.

When gathering requirements, these diagrams offer two main advantages: 1) they are able to provide a detailed description of each requirement, and the role they fulfill within the system. 2) They define with detail the functionality and interaction of the

subsystems that compose the system. While a proper software requirements specification (SRS) document is still in order, a requirement diagram expands everything stated on the SRS and supports it with an extensive graphical definition.

While requirements diagrams are a central part of SysML models, they lose their value if not supported by other diagrams to expand on what has been defined on the main model. These supporting diagrams are divided into behavior and structure diagrams.

### 3.1 Behavior Diagrams

Behavior diagrams are broken down into four types: activity, sequence, state machine and use case diagrams.

Activity diagrams specify the necessary process to transform inputs into outputs. This diagram represents the flow of information from an initial state to a final one. These flows traverse several different activities and they're able to fork into separate paths, or converge into a single one in any given moment. It's useful because it allows for conditions to be established and then to analyze how each of them affect the system. A single activity can also invoke an entirely new activity diagram with its own action flow to produce an output for said activity. An example of this diagram can be seen in Figure 9.

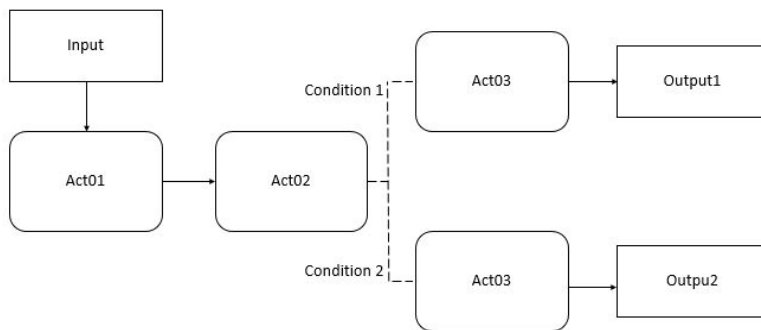


Figure 9. Activity Diagram Example

Sequence diagrams are designed to represent interactions among different subsystems or actors. They provide tools that allow for the creation of complex scenarios, including logical routing, system decomposition and referencing sequences. Its main advantage is that it models an overview of the average behavior of the system, and how it will interact with external agents.

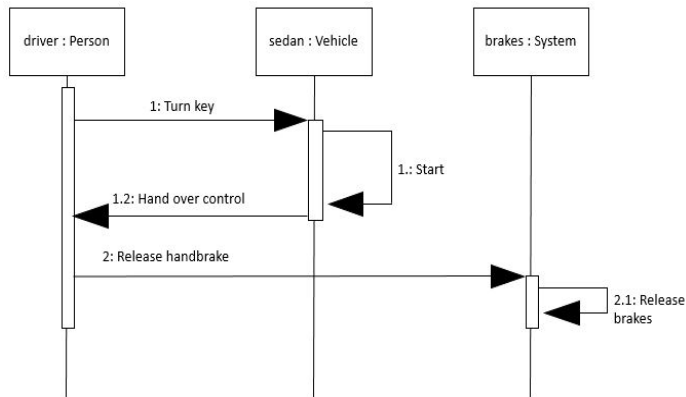


Figure 10. Sequence Diagram Example

These diagrams show all actors involved with the system as white boxes with a lifeline extending beneath them: users, interfacing systems and the system itself. These lifelines are connected by lines whenever an actor interacts with another, and each line is usually preceded by a precondition, the line itself has the action occurring, and the receiving actor has the outcome of said action.

State machine diagrams are used to represent the lifecycle of a system block. They are used because they support asynchronous event-based behavior, stating triggers, guards, actions, entries, exits and if-then activities. They can include nested state diagrams and permit the communication between blocks during transitions. Figure 11 shows a simple example of a state machine diagram.

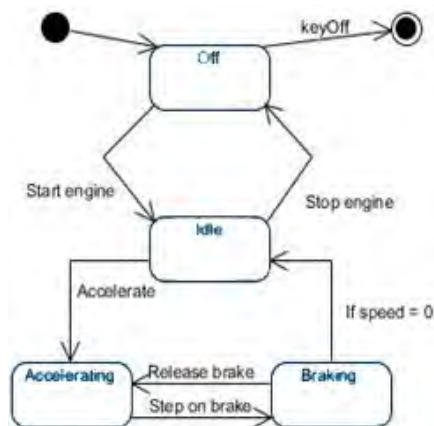


Figure 11. State Machine Diagram Example

One of the most used diagrams in software development, use case diagrams (see Figure 12), describe basic interactions between system actors and the goals expected from them. Relationships between use cases can be noted by the `<<include>>` and `<<extend>>` tags.

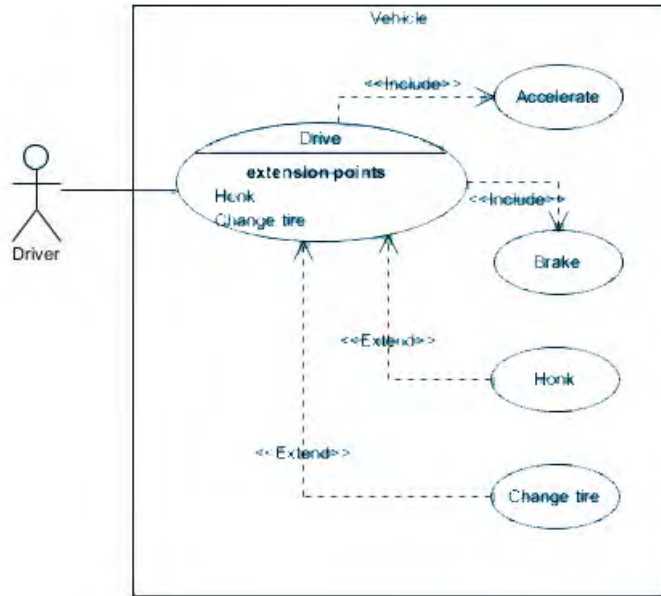


Figure 12. Use Case Diagram Example

Include relationships are used when an entire use case's functionality is going to be used in another one. Extend relationships are used when the behavior described in a use case can be, optionally, included in another use case.

As stated before, SysML is a modified version of UML. While activity diagrams have been adapted from their UML form, sequence, state machine and use case diagrams present virtually little to no change from their UML counterparts.

## 3.2 Structural Diagrams

Structure diagrams are separated in block definition, internal block, package and parametric diagrams.

Package diagrams are the only structural diagrams to be unchanged from UML. They are used to organize the whole model by assigning group elements into a name space. The

model can be organized in several different ways: by system hierarchy, diagram type or viewpoints to augment the organization. An example of a package diagram is shown in Figure 13.

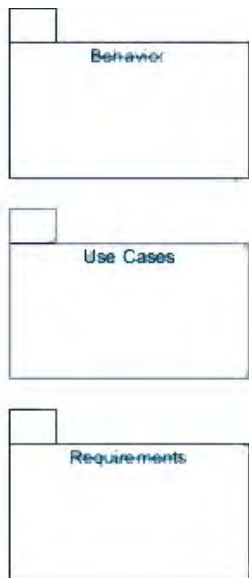


Figure 13. Package Diagram Example

Block definition diagrams describe the overall structure of an element or even a complete system. They are represented as a box with multiple compartments used to describe characteristics such as properties, operations, constraints, allocations, requirements satisfied or any other user defined compartment.

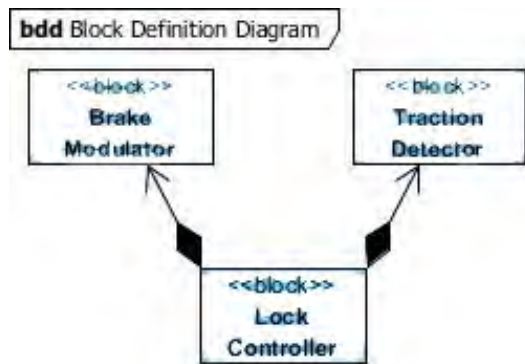


Figure 14. Block Definition Diagram Example

As seen in Figure 14, block definition diagrams establish the connections and relationships between multiple blocks within a system.

Each block can be further refined by an internal block diagram. These describe the internal structure of a block in relation to its properties and connectors. They show parts of the block, connected by lines and specifying the item flow between them. This can be seen in Figure 15.

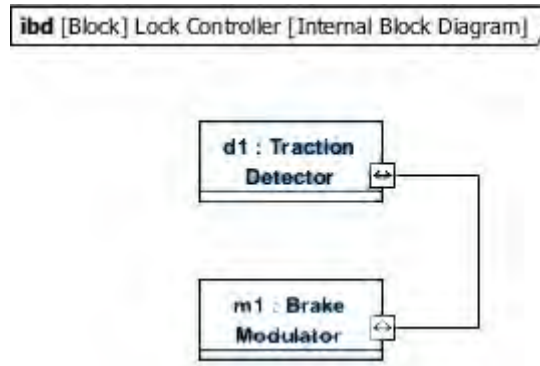


Figure 15. Internal Block Diagram Example

A new diagram introduced by SysML is the parametric diagram. It is used to express constraints between value properties. This provides support for engineering analyses and facilitates the identification of critical performance properties. These can be expressed in a formal or informal language. Constraints are applied to a block and allow the designer to identify scenarios to take into account when developing the block. Figure 16 shows a simple parametrics diagram.

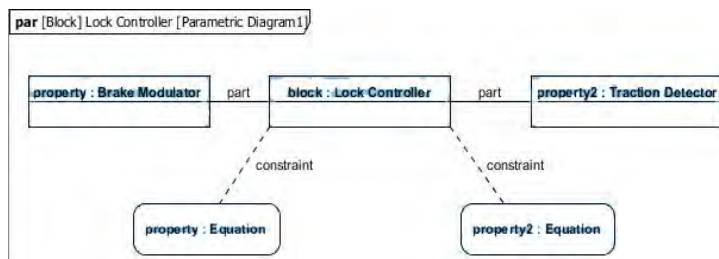


Figure 16. Parametrics Diagram Example

Overall, the interaction between structural diagrams goes: internal block diagrams define a block while at the same time a parametrics diagram specifies the constraints it

involves. Then, blocks are connected in a block definition diagram before being grouped up in its own package diagram.

## 4. Conclusions

Software development is definitely not a simple task. If not taken seriously, defective software systems will be delivered to clients, resulting in economic losses or even worse. As said by Charette [2] “we waste billions of dollars each year on entirely preventable mistakes”. Requirements are only the first step towards quality software, but it just might be the most crucial one. To keep in line with ISO 26262, requirements should be modeled in a semi-formal manner. SysML provides the necessary tools required to do exactly that. Detailed requirements specification provides an in-depth look to the system’s behavior, and therefore, helps with the following phases of software development. This in turn, ensures the correct functioning of the software and prevents future monetary losses or even, human casualties.

While there is no evidence to prove that SysML modeling reduces the amount of defects found when developing a system, we believe that, by providing designers and developers with in-depth descriptions of every single component that integrates an automotive software system, the crucial planning and design phases would become more robust. In this scenario, by performing a thorough modeling of the system, features would not be overlooked, points of failure would be detected and not so many defects would be injected into the system.

## 5. References

- [1] IEEE, IEEE Std. 610.12-1990, “IEEE Standard Glossary of Software Engineering Terminology”, 2002, Institute of Electrical and Electronics Engineers.
- [2] R. Charette, “Why Software Fails”, Sept-2005, IEEE Spectrum. [Online]. Available: <http://spectrum.ieee.org/computing/software/why-software-fails/0>
- [3] L. Geppert, “Lost Radio Contact Leaves Pilots On Their Own”, Nov-2004, IEEE Spectrum. [Online]. Available: <http://spectrum.ieee.org/aerospace/aviation/lost-radio-contact-leaves-pilots-on-their-own>
- [4] N. Bashar, “Ariane 5: Who Dunnit?” May-1997, IEEE Software.
- [5] D. Firesmith, “Common Requirements Problems, Their Negative Consequences, and the Industry Best Practices to Help Solve Them”, Feb-2007, Journal of Object Technology. [Online]. Available: [http://www.jot.fm/issues/issue\\_2007\\_01/column2.pdf](http://www.jot.fm/issues/issue_2007_01/column2.pdf)
- [6] C. Areias, J. Cunha, D. Iacono, F. Rossi, “Towards Certification of Automotive Software”, 2014 IEEE International Symposium on Software Reliability Engineering Workshops.



- [7] Broy, M., Kirstan, S., Krcmar, H., and Schätz, B, "What is the benefit of a model-based design of embedded software systems in the car industry?", in Rech, J., and Bunse, C. (Hrsg.) *Emerging Technologies for the Evolution and Maintenance of Software Models*. 2011. 343-369.
- [8] A. Mjeda, M. Hinchey, "Requirement-Centric Reactive Testing for Safety-Related Automotive Software", 2015 IEEE/ACM 2<sup>nd</sup> International Workshop on Requirements Engineering and Testing.
- [9] M. Aoyama, A. Yoshino, "AORE (Aspect-Oriented Requirements Engineering) Methodology for Automotive Software Product Lines", 2008 15<sup>th</sup> Asia-Pacific Software Engineering Conference.
- [10] J. Hartmann, S. Rittmann, D. Wild, P. Scholz, "Formal Incremental Requirements Specification of Service-Oriented Automotive Software Systems", 2006 IEEE Computer Society.
- [11] X. Liu, Z. Wang, "Extending EAST-ADL2 to Support Aspectual Requirement Specification and Analysis for Automotive Software", 2011 International Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST-11.
- [12] X. Liu, X. Yan, Y. Li, X. Che, C. Mao, "Modeling Automotive Software Requirements with Aspectual Models", 2010 Second WRI World Congress on Software Engineering.
- [13] ISO, ISO 26262, "Road vehicles – Functional Safety". Part 8: "Supporting Processes". 2011. International Organization for Standardization.
- [14] G. Lami, F. Falcini, "Automotive SPICE Assessments in Safety Contexts: an Experience Report", 2014 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW).
- [15] J. Cobb, "IBM outlines how it helped Chevrolet with the Volt's development", [On line]. Available <http://gm-volt.com/2011/05/11/ibm-highlights-its-help-with-the-chevrolet-volts-development/>
- [16] D.Zax,"ManyCarshaveahundredmillionlinesofcode",[Online].Available <https://www.technologyreview.com/s/508231/many-cars-have-a-hundred-million-lines-of-code/>
- [17] H. G. Chalé Góngora, O. Taofifenua, T. Caudré, A process and Model for Automotive Safety-Critical Systems Design, INCOSE, Vol. 20, Issue 1, pp 1211-1226. 2010. DOI: 10.1002/j.2334-5837.2010.tb01135.x
- [18] ISO, ISO 26262 "Road vehicles – Functional safety". Part 9: "Automotive Safety Integrity Level (ASIL)-oriented and safety-oriented analyses". 2011, International Organization for Standardization
- [19] EAST – ADL Domain Model Specification V2.1.12, 2011-2013, EAST – ADL Association.
- [20] OMG Systems Modeling Language (OMG SysML), Version 1.4, 2015.
- [21] P. Filipovikj, M. Nyberg, G. Rodríguez-Navas, "Reassessing the Pattern-Based Approach for Formalizing Requirements in the Automotive Domain", 22<sup>nd</sup> International Requirements –Engineering Conference (RE), 2014. P. 444-450.

- [22] M. H. L. Lee, W. C. In, "Informal, Semi-formal, and Formal Approaches to the Specification of Software Requirements", Thesis Angelo State University. 1992
- [23] D. C. Ince, "An introduction to Discrete Mathematics, Formal System Specification, and Z", Oxford University Press, Inc., New York, 1992.
- [24] E. Gulias, L. F. Torreblanca, J.R. Aguilar, C. Fernandez, "Using SysML Modeling to Accurately Represent Automotive Safety Requirements", 4<sup>th</sup> International Conference on in Software Engineering Research and Innovations. CONISOF 16, IEEE. DOI 10.1109/CONISOFT.2016.12.

# Chapter # 14

## Conceptual synthesis of practice as a theoretical construct in Software Engineering

Alexander Barón-Salazar  
Galeras.Net Research Group  
Universidad de Nariño  
San Juan de Pasto, Colombia  
abaron\_98@udenar.edu.co

Carlos Mario Zapata-Jaramillo  
Research group in Computer  
Languages  
Universidad Nacional de Colombia—  
Sede Medellín  
Medellín, Colombia  
cmzapata@unal.edu.co

### 1. Introduction

Semat (Software Engineering Method and Theory) seeks to redefine software engineering from the critical issues the community identifies. Semat focuses on two main objectives: finding a widely accepted kernel of elements describing the essence of software engineering, and defining an appropriate and widely accepted theoretical basis for the discipline [1].

Any process to develop a unified theoretical basis should include a parallel effort to achieve consensus on terminology. This task is difficult for software engineering, since a wide range of definitions and interpretations of commonly used terms in the profession are used [2]. According to Johnson and Ekstedt [3], software engineering lacks a common vocabulary to efficiently share knowledge, and they propose the definition of this vocabulary as the first step towards building a general theory.

A conceptual synthesis is a way to build a unified definition of a theoretical construct with several definitions in the same context. Such a synthesis allows for identifying common elements about the way how several proposals characterize and define such term.

Software practice is a theoretical construct with several definitions from the perspective of each proposal. Some approaches define and characterize software practice in detail

[1, 4, 5]; others define it from specific contexts [6]; in other cases, no explicit definition is done [7]. Summarizing, no definition reconciles the different points of view, so a conceptual reference for the construct of “practice” in the software engineering domain is still needed.

In this Chapter, we propose a conceptual synthesis of “practice” as a theoretical construct in software engineering by using a strategy for easing the identification, collection, and analysis of the relevant state of the art. The applied strategy is based on the process of Systematic Literature Review (SLR) proposed by Kitchenham and Charters [8].

SLR should be thorough and unbiased, otherwise, a scientific value is unachieved; SLR is thorough when it includes all the relevant state-of-the-art; SLR is unbiased when it includes enough of the state-of-the-art review for supporting and refuting the hypothesis [9]. Thus, the strategy we apply allows for a conceptual synthesis for ensuring valid results as a basis for a definition of the practice as a theoretical construct the software engineering community can accept and share.

Our methodological approach is based on pre-conceptual schemas as a mechanism for the extraction and synthesis of information. A pre-conceptual schema is a representation of a specific domain. Pre-conceptual schemas allow for representing the terminology of a domain in order to ease their translation into conceptual schemas [10].

We structure of this Chapter as follows: in Section II we describe the theoretical framework; In Section III, we present the background on how different approaches define software engineering practice; in Section IV we describe the methodological approach for the development of conceptual synthesis in software engineering; in Section V we propose the conceptual synthesis of practice as a theoretical construct in software engineering; finally, conclusions and future work are discussed in Section 5.

## **2. Theoretical Background**

### **2.1 Kernel for methods in software engineering**

The call for action is a statement of the community for describing software engineering as a discipline with immature practices and specific problems [11]. Semat leads the development and promotion of a kernel and a language for defining methods and software engineering practices in order to address the software engineering problems [12].

The Semat kernel provides a common ground for the definition of software development practices and includes the essential elements prevailing in each software engineering endeavor. The kernel allows for composing practices and creating specific methods which can be tailored to the particular needs of a community, a project, a team, or an organization. So far, OMG (Object Management Group) has published two versions of the standard Essence–Kernel and Language for Software Engineering Methods–[5].

### 2.1.1 Areas of concern

Semat kernel is organized in three areas of concern: customer, solution, and endeavor. Each area of concern is focused on a specific aspect of software engineering. Customer area of concern contains everything related to the use and operation of the software system; solution area of concern contains everything people need to specify and develop the software system; finally, endeavor area of concern contains everything related to the team and how the team does its work. Each area of concern contains alphas, activity spaces, and competencies. The areas of concern of the Semat kernel are shown in Figure 1.

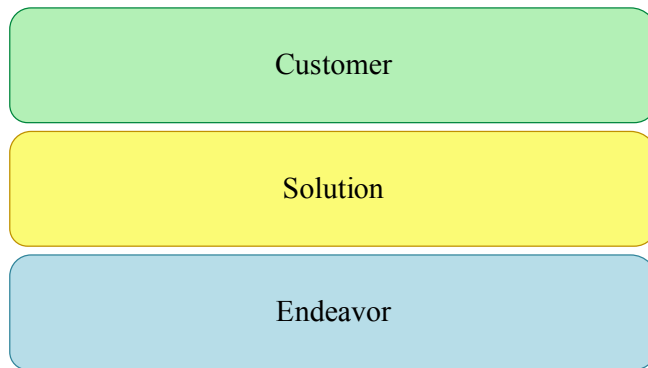


Figure 1. Areas of concern of the kernel [1].

### 2.1.2 Alphas

An Alpha—Abstract-Level Progress Health Attribute—captures the key concepts involved in software engineering. Each Alpha has a set of predefined states used to assess the health and progress of a software engineering endeavor; also, each state contains a checklist to verify compliance with the conditions of the state.

In the Customer area of concern, the team needs to understand stakeholders and the opportunity of a software engineering endeavor. Opportunity Alpha is the set of circum-

stances that make up an appropriate scenario to develop or change a software system. Stakeholders Alpha constitutes individuals, groups or organizations that affect or are affected by a software system.

In the Solution area of concern, the team needs to establish a common understanding of the requirements to implement, build, test, deploy, and maintain a software system satisfying such requirements. Requirements Alpha is what the software system has to do to treat the opportunity and satisfy stakeholders. Software System Alpha includes software, hardware, and data.

Team, work, and way of working are defined in the Endeavor area of concern. Work Alpha involves a physical or mental endeavor to get a result. Team Alpha is a group of people actively involved in the development, maintenance, deployment, and support of a specific software system. Way of Working Alpha is the set of practices and tools adapted that a team uses to guide and support its work. Alphas of the Semat kernel are shown in Figure 2.

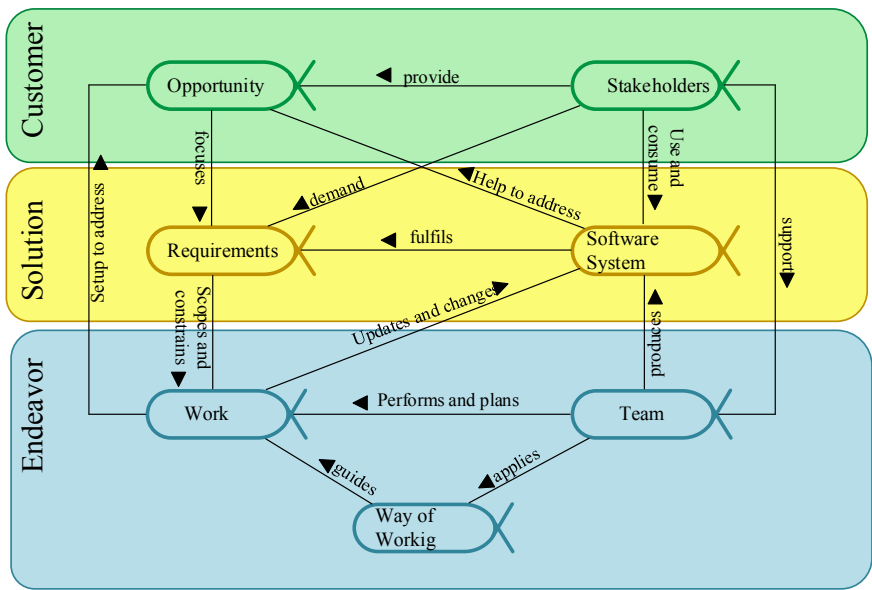


Figure 2. Alphas of the kernel [1].

### 2.1.3 Activity Spaces: The things to do

The kernel has a set of activity spaces to provide a vision based on the activity of software engineering.

In the Customer area of concern, the team must understand the opportunity and engage stakeholders. The activity spaces of the Customer area of concern are: explore possibilities, understand the stakeholder needs, ensure the stakeholder satisfaction, and use the system.

In the Solution area of concern, the team should develop an appropriate solution to address the opportunity and meet requirements. The activity spaces of the Solution area of concern are: understand the requirements, shape the system, implement the system, test the system, deploy the system, and operate the system.

The team is formed in the Endeavor area of concern, and the progress of work is aligned with the way of working. The way of working depends on the team limitations and governance rules. The activity spaces of the Endeavor area of concern are: prepare to do the work, coordinate activities, support the team, track progress, and stop the work. Activity spaces of the Semat kernel are shown in Figure 3.

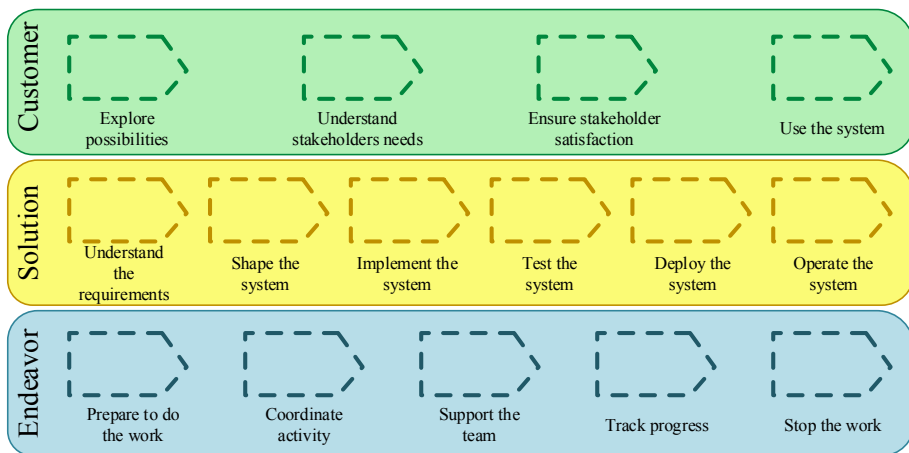


Figure 3. Activity Spaces [1].

## 2.1.4 Competencies: The Abilities Needed

The kernel provides a set of skills complementing the alphas and activity spaces to provide an overview of the key skills required to perform the work of a software engineering endeavor.

In the Customer area of concern, the team must be able to demonstrate a clear understanding of the technical aspects and the business, and the team should have the ability

to accurately convey the viewpoint of the stakeholders. The Customer area of concern requires competencies for representing the stakeholders.

In the Solution area of concern, the team has to be able to capture and analyze the requirements, and build and operate a software system satisfying such requirements. So, the following competencies are needed: analysis, development, and testing.

In the Endeavor area of concern, the team has to be able to organize and manage work. Accordingly, some team members should have leadership and management skills. Competencies of the Semat kernel are shown in Figure 4.

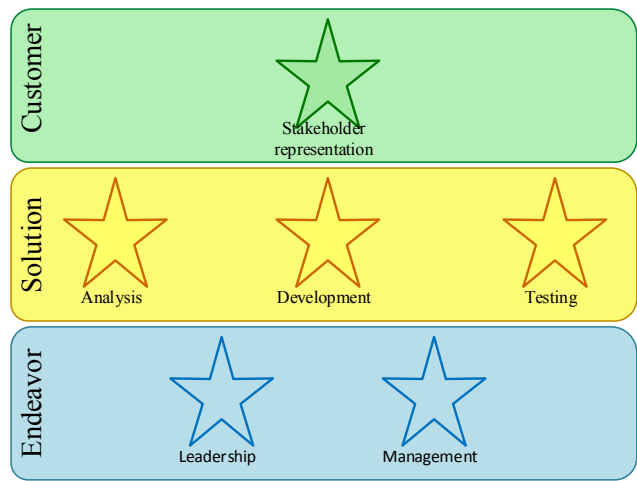


Figure 4. Competencies of the kernel [1].

## 2.2 Systematic Literature Review—SLR

An SLR is intended to summarize, compile, and synthesize existing research on a subject area or phenomenon of interest. SLR is a contribution to current knowledge, because its findings are obtained when the relevant state of the art is analyzed as a whole instead of isolated reading documents. SLR should classify the state of the art, identify research trends, support for further research, and establish the importance of a research problem [14].

### 2.2.1 Some work on SLR

Lenberg et al. [15] show an SLR about the human behavior in software engineering. Such study has the aim to show the interest of the scientific community about the human as-



pects of software engineering and create a common platform for future research in the area. The authors propose a definition of behavior in software engineering and present the results of an SLR based on this definition. Some other studies show the increasing application of SLR in software engineering [16, 17, 18, 19].

### 2.2.2 Kitchenham and Charters proposal [8]

A methodological strategy for the development of conceptual syntheses is based on a guide for conducting an SLR in software engineering [8]. Such a guide presents a set of lessons on the implementation of the process of SLR in the software engineering domain [20, 21] and the results of an SLR directed to identify, evaluate, and synthesize published research on experiences of systematic reviews and proposals to improve the process [9]. According to Kitchenham and Charters [8], an SLR involves several discrete activities grouped into three phases: planning, realization, and reporting. Process for conducting SLR is illustrated in Figure 5.

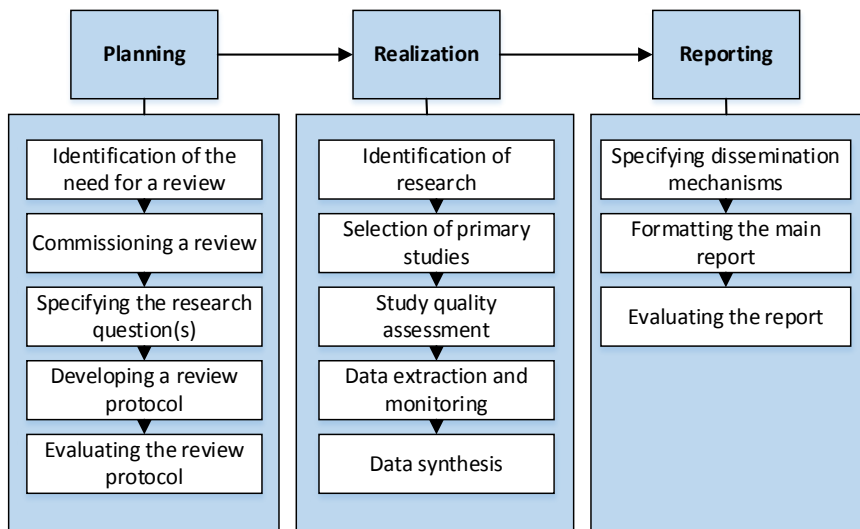


Figure 5. The process for SLR [8].

## 2.3 Pre-conceptual schema

Pre-conceptual schemas are representations of a specific domain the experts can validate. Pre-conceptual schemas allow for the representation of the terminology of a domain for easing their translation into conceptual schemas. Pre-conceptual schemas use a notation based on conceptual graphs, with additional symbols representing dynamic

properties. A pre-conceptual schema is a labeled, unlooped di-graph with multiple arcs, made with nodes that are connected with arches [10]. The basic syntax of pre-conceptual schemas is shown in Figure 6.

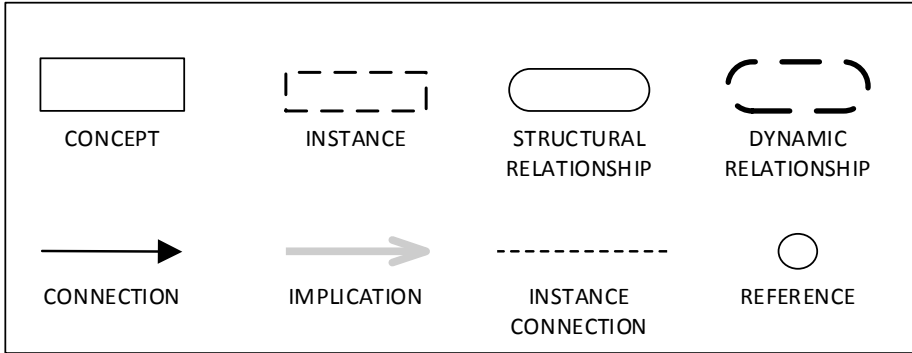


Figure 6. Basic syntax of pre-conceptual schemas [10]

### 3. Practice in software engineering

Kirk and Tempero [4] try to better understand how successful organizations adapt software practices according to their context and their objectives to the software, based on the premise: “the best practice depends on the application context.” They argue the software development activities are the same in all organizations, but the difference lies in the way how they are performed. They conceive software practice as the strategy for addressing a development activity, and they warn the success of the activity largely depends on the adaptation of practice to the specific context of the project.

The research of Rolandsson et al. [7] describes a study of companies with two software development approaches: proprietary software (closed work practices) and open source software (based on cooperation and shared knowledge practices). This approach integration results in transforming their practices: practices of open source software are documented and implemented by controls; instead, proprietary software practices associate values such as helping others, autonomous learning, and voluntary cooperation. The study lacks an explicit definition of practice of software, but we can implicitly infer a practice as an activity of the development process, which is common to both approaches. Such software practice refers to how the development team interacts according to the philosophy of the approach. In this case, the definition focuses on research interest: the ease of integration of the practice.

Passos et al. [6] study the software practice from an ethnographic approach. Ethnography related to the adoption of a cultural perspective for observing and interpreting the events, actions and behaviors within a specific community. The results show ethnographic methods are indispensable for understanding the software practice. They conceive software practice as a software activity with cultural, social, and political aspects influencing the context. In this case, the proposal focuses on the definition of the software practice from human aspects, regardless of other visions.

The Object Management Group defines a specification about the Kernel and Language for Software Engineering Methods proposed by the Semat (Software Engineering Method and Theory) Community; such Kernel and Language are known as Essence [5]. Practice is defined by the OMG as a repeatable approach for a specific purpose. A practice provides a systematic and verifiable manner to address a particular aspect of a work. The practice has a clear objective expressed in terms of the results that can be applied; the practice provides guidance to help and guide practitioners on what should be done to achieve the goal, ensuring that the objective is understood in order to be verified when such objective is achieved.

OMG exemplifies his proposal by using Scrum and user stories. In the same specification, a language extension is applied to the Daily Scrum Meeting as software practice; meanwhile, software practice is presented in another section of the specification as an activity [5].

Jones [22] makes a compilation of best practices belonging to several technical areas of software engineering. He proposes some criteria for determining when a practice acquires the status of best practice and the measurement method to evaluate it. The author believes that to be classified a practice as a best one, a language, a tool or a method should exhibit some quantitative evidence in terms of improving quality, productivity and other tangible factors.

Capability Maturity Model Integration (CMMI) represents collections of best practices for supporting organizations in their improvement process. The model for developing CMMI-DEV is a collection of best practices covering the product lifecycle from conception to delivery and maintenance. Levels are used in CMMI-DEV to describe improving product development processes or services. A level comprises a set of process areas, and a process area is a set of generic and specific practices. Generic practices describe the activities considered important for a generic goal and contribute to the institutionalization of the process associated with a process area. A specific practice is the description of an activity that is considered important to achieve

a specific objective of a process area [21]. The definition of practice the model represents is goal-oriented.

Scrum is a management framework used in agile projects in order to iteratively deliver increments of high customer value. Projects progress through a series of iterations called Sprints in Scrum. The work to be done in a Scrum project is listed in the product backlog of the project. At the beginning of each Sprint, a planning meeting in which the product owner prioritizes the work of the product backlog is made; then the Scrum team selects the tasks that can be delivered during the next Sprint. Such tasks are moved from the product backlog of the project to the product backlog of the Sprint. Each day during the sprint is held a brief meeting, which helps the team to stay aware of the evolution of the project. At the end of each Sprint, the team demonstrates the full functionality at a Sprint Review Meeting. Other terms than practice are used for describing the Scrum framework; however, 19 processes integrated into four phases are described in Scrum. Each phase describes the process in detail including inputs, tools and associated outputs. In each process, some elements are mandatory and other ones are specific to the project, organization or industry [22].

Rational Unified Process (RUP) is an iterative, incremental software engineering process, focusing on architecture and directed by use cases. RUP is aimed to ensure the production of high quality software meeting the needs of its end users, within a predictable schedule and budget [23]. RUP provides guidelines, templates, and tools needed to effectively implement best practices for solving common problems in software projects. RUP is described in two dimensions: horizontal (dynamic aspects) and vertical (static aspects). The dynamic aspects are cycles, phases, iterations, and milestones; the static aspects are activities, artifacts, actors, and workflows. In RUP, a best practice is a way to address an activity of the software process which is tested in real contexts. RUP practice runs on the dynamic dimension and includes elements of the static dimension [24].

Eclipse Process Framework Composer (EPFC) is a tool for software process management in organizations. EPFC allows for defining processes by using software practices integrated in methods. EPFC defines practice as a documented approach for solving one or more recurrent problems. Practice in EPFC is associated with work products generating practices; tasks defining the steps of practice development; guidelines indicating the way of applying the practice and the roles played by participants in the practice [25].

Some other work incorporates elements according to the research interest to the software practice concept: Barón [27] presents the software practice as an asset of knowled-

ge; García et al. [28] promote the use of software patterns; Zhang *et al.* [29] suggest the application of simulation processes; Torkar *et al.* [30] present strategies to adopt industry practices, techniques and methods of open source software; Meso and Jain [31] present a set of principles and best practices for adaptive systems, from the perspective of agile software development.

## 4. Methodological approach for creating conceptual syntheses

The SLR process of Kitchenham and Charters [8] is the basis of the methodological approach for the development of our conceptual synthesis. Phases of the methodological approach [26] are shown in Figure 7.

### 4.1.1 Planning phase

Activity spaces, activities and work products of the planning phase are shown in Figure 8.

#### 1. Identification of the needs of the conceptual synthesis

The need for an SLR arises from the researcher requirements to summarize all existing information on a phenomenon thoroughly and impartially [26].

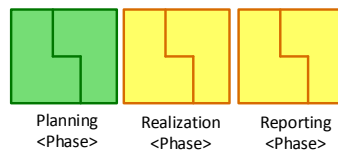


Figure 7. Phases of the methodological strategy.

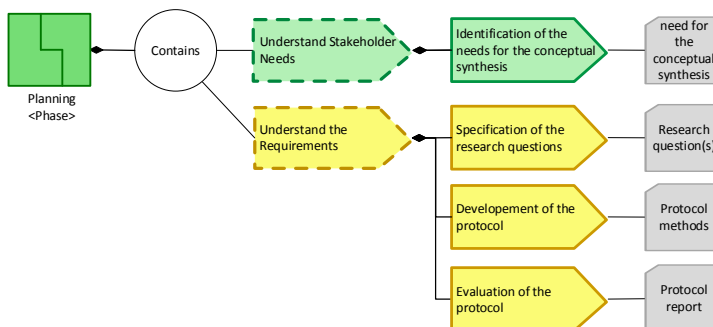


Figure 8. Planning phase.

## 2. Hiring the review

Sometimes, an organization requires information about a specific topic, but the organization lacks the time or the expertise to perform an SLR. In such cases, the organization engages the services of researchers to conduct the study. In this regard, representatives of the organization and researchers need to define the requirements of the SLR [26].

## 3. Specification of the research questions

Questions guide the search for activities of primary studies by using extraction and synthesis of information to answer the questions [26].

## 4. Development of the protocol

During the development of the protocol, researchers specify the methods to be used to conduct an SLR. The protocol should specify methods to ensure a thorough and impartial SLR. [26].

## 5. Evaluation of the protocol

The protocol is a critical element of any SLR. Researchers should agree on a procedure for evaluating the protocol [8]. For the development of conceptual syntheses in software engineering, the strategy proposes the implementation of the protocol to a sample of relevant papers [26].

### 4.1.2 Realization phase

Activity spaces, activities and work products of the realization phase are shown in Figure 9.

#### 1. Identification of the research.

A thorough and impartial search strategy is required to find the greatest number of primary studies related to the research questions [8]. The methodological approach is intended to specify search strings for identifying the research. Search strings are configurations describing the research questions. Such strings are the criteria entered in the search engines of digital sources [26].

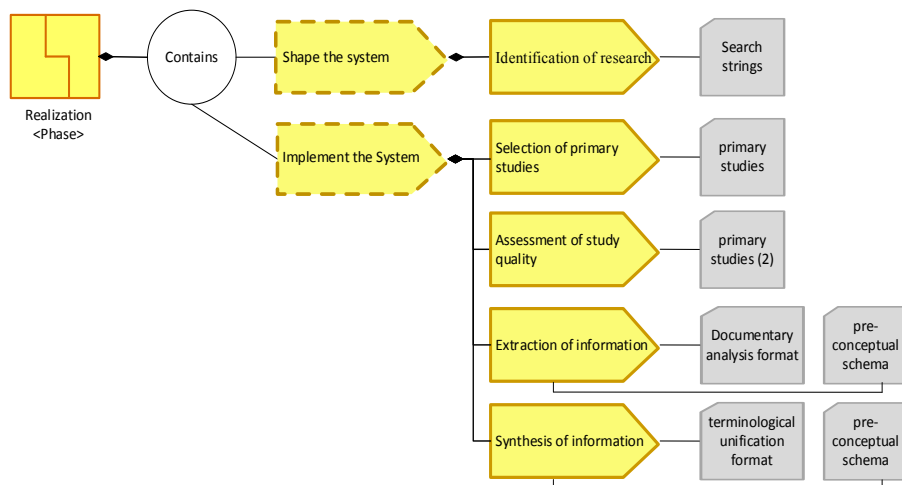


Figure 9. Realization phase.

## 2. Selection of primary studies

This activity is aimed to select the actual relevant studies contributing to answer the research questions [8].

### a. Identification of the sources of studies

The sources of the state of the art about the research topic are determined at this point. Currently, digital sources are frequently used as repositories of relevant studies. Also, some studies recognized by the community should be included, even though they can be excluded by digital sources [26].

### b. Selection of studies

Some studies are evaluated based on their actual relevance by using search strings in digital sources. The selection criteria are used to identify studies that provide direct evidence on the research question. The selection criteria are defined according to the research question [8]. We use in our methodological approach the use of inclusion and exclusion criteria as proposed by Kitchenham and Charters [8]. Inclusion criteria allow for identifying the relevant state of the art of the study, while exclusion criteria allow for identifying irrelevant studies to be omitted [26]. This process is performed in two iterations including: i) the title of the study and ii) the abstract and the keywords.

### 3. Assessment of study quality

Additional criteria for assessing the quality of primary studies should be considered [8]. We propose some activities after extracting and synthesizing the information in order to gain a more detailed view of the study. This detailed view allows researchers for determining the inclusion or exclusion of the final concepts to the synthesis. In our case, such criteria are applied as additional filters to avoid bias and ensure the inclusion of actually relevant studies [26].

### 4. Extraction of information

This phase is aimed to design data extraction forms to accurately record the information the researchers obtained from the selected studies [8]. In our approach, each of the relevant studies is subjected to a process of analysis. The analytical products of the study are the terminological unification and the pre-conceptual schema. Terminological unification is related to the integration of the terms referred to the same concept in a common term; on the other hand, a pre-conceptual schema summarizes how the study defines the theoretical construct of software practice [26].

### 5. Synthesis of information

In this phase we integrate and summarize the results of the relevant studies. The synthesis can be descriptive and supplemented with a quantitative summary [8]. In our approach, inputs are the formats of documentary analysis of relevant studies. Terminological unification is consolidated by using such inputs and the pre-conceptual schema is drawn as a preliminary version of the theoretical construct of software practice [26].

#### 4.1.3 Reporting phase

In this final phase of an SLR, we write the results of the review and communicate the results to the community concerned. [8]. Activity spaces, activities and work products of the report phase are shown in Figure 10 [26].

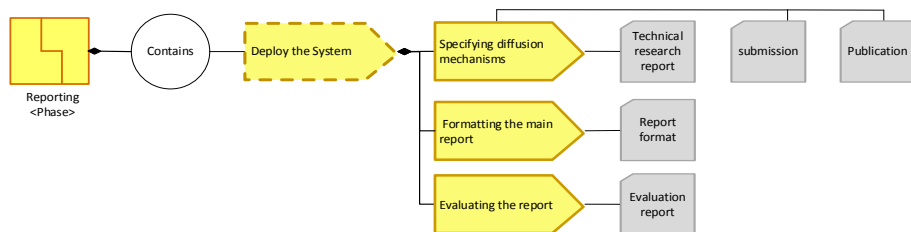


Figure 10. Reporting phase.



## 1. Specifying diffusion mechanism

Diffusion of the results of an SLR should be effective. For this reason, most guidelines recommend planning strategy for the diffusion [8]. In our approach we suggest specifying communication media according to the specificity of the conceptual synthesis [26].

## 2. Formatting the main report

Typically, communication media constrain format and length report. However, some structure and scope of the report are suggested in order to ensure efficiency of communication to the community concerned [8]. In our approach we use some formats of the guide [8] and the formats established in journals and academic events, in which the results of the conceptual synthesis should be presented [26].

## 3. Evaluating the report

The Main report is evaluated according to the communication media. A paper submitted to a journal or an event is evaluated by scientific committees and the target community in terms of rigor and validity of the SLR. When the report is part of a Ph.D. Thesis, the SRL is evaluated in the overall evaluation of the research project [8].

# 5. Conceptual synthesis of the software practice

## 5.1 In this Section we apply our approach to the theoretical construct of practice in software engineering.

### 5.1.1 Planning phase

Identification of the needs of the conceptual synthesis

We justify the conceptual synthesis of the theoretical construct of practice in software engineering as follows: the conceptual synthesis aims to identify common aspects in the definition of the theoretical construct of practice within the context of software engineering. The results will be the input for characterizing the theoretical construct, and they will be the input for formulating a unified definition. This definition is a contribution to

achieve one of the goals of the Semat initiative: defining an adequate and widely accepted theoretical basis for software engineering.

### **5.1.2 Hiring review**

Conceptual synthesis of the theoretical construct of software practice is the input for the development of future research, and especially for the construction of a unified definition. Researchers interested have the expertise required to conduct the study; therefore, we need no contracts to develop the conceptual synthesis.

### **5.1.3 Specification of the research questions**

The research question is: How the software engineering proposals define the theoretical construct of practice?

### **5.1.4 Development of the protocol**

The methods of the strategy for the development of conceptual synthesis are defined and applied according to the SLR process.

### **5.1.5 Evaluation of the protocol**

In the case of the theoretical construct of software engineering practice, the protocol was applied to Kuali Beh [5] and CMMI-DEV [22]. Such samples allow for adjusting the protocol to subsequent application to the universe of the relevant state of the art.

## **5.2 Realization phase**

### **5.2.1 Identification of the research**

Search strings defined in the case of the conceptual synthesis construct of practice are shown in Figure 11.

**String 1:** “Software practice” or “software development practice” or “Software engineering practice”

**String 2:** “Software method” or “software development method” or “Software engineering method”

**String 3** “Software Theory” or “software development theory” or “Software engineering theory”

**String 1:** “Software practice” or “software development practice” or “Software engineering practice”

**String 2:** “Software method” or “software development method” or “Software engineering method”

**String 3** “Software Theory” or “software development theory” or “Software engineering theory”

Figure 11. Search strings

## 5.2.2 Selection of primary studies

### a. Identification of sources of studies

The search for the state of the art about the theoretical construct of practice in software engineering is done by using digital sources, such as: ACM Digital Library [23], EBSCO [24], Engineering Village [25], IEEE Xplore Digital Library [26], ScienceDirect [27], Scopus [28] and Web of science [29]. Similarly, some other studies are recognized by the software engineering community and digital sources and included in this identification process. The results of this exercise are shown in Table 1.

Table 1. Search results in digital sources

Source of studies	Selected studies			
	String 1	String 2	String 3	Total
ACM Digital Library	139	64	8	211
EBSCO	78	73	2	153
Engineering Village	166	235	52	453
IEE Xplore Digital Library	101	81	21	203
ScienceDirect	54	113	10	177
Scopus	491	716	84	1291
Web of Science	37	61	8	106
Total	1066	1343	185	2594

### b. Selection of studies

Inclusion criteria of the conceptual synthesis of the theoretical construct of software practice are studies related to software engineering practices, software engineering

methods, and software engineering theories. Exclusion criteria used in the conceptual synthesis of the theoretical construct of software practice are studies related to teaching practices, teaching methods, and theories of other disciplines than software engineering.

This process is performed in two iterations considering: i) the title of the study; and ii) abstract and keywords. The results of applying the inclusion and exclusion criteria are shown in Table 2.

**Table 2. Results of the application of the inclusion/exclusion criteria**

Source of studies	Selected studies			Total
	String 1	String 2	String 3	
ACM Digital Library	11	4	1	16
EBSCO	6	1	0	7
Engineering Village	18	21	10	49
IEE Xplore Digital Library	12	8	4	24
ScienceDirect	10	8	3	21
Scopus	60	64	12	136
Web of Science	9	7	1	17
Total	126	113	31	270

### 5.2.3 Assessment of study quality

In our case, such criteria are applied as additional filters to avoid bias and ensure the inclusion of actually relevant studies.

### 5.2.4 Extraction of information

Examples of pre-conceptual schemas resulting from the extraction of information are presented in Figures 12 and 13.

### 5.2.5 Synthesis of information

We include some proposals in the conceptual synthesis for establishing a preliminary version of the definition of the theoretical construct of software practice. This preliminary version integrates four views: static, operational, human, and management view. Overview and views are shown in Figures 14 to 18.

## 5.3 Reporting phase

### 5.3.1 Specifying diffusion mechanisms

In this case, the conceptual synthesis is written as a technical research report and as a Section of a Ph.D. Thesis. Diffusion is done by submitting papers to journals, and the development of presentations in academic events related to software engineering.

### 5.3.2 Formatting the main report

The methodological strategy suggests using some formats of the guide [8] and the formats established in publications and academic events, which present the results of the conceptual synthesis.

### 5.3.3 Evaluating the report

The conceptual synthesis of the theoretical construct of practice is part of a Ph.D. Thesis. Therefore, the software engineering community—including the academic research process and events—is in charge of evaluating the results.

## 6. Conclusions and future work

In this Chapter we proposed to the academic community a methodological approach for the development of conceptual synthesis in software engineering and we applied it to the theoretical construct of software practice. Kitchenham and Charters [8] propose to use an SLR as a guide for defining the methodological strategy, since they provide the process and the appropriate methods for conceptual synthesis. The exercise allows for demonstrating the benefits of scalability, flexibility and adaptation of the proposal by Kitchenham and Charters [8]; in addition, we incorporate pre-conceptual schemas to the synthesis in order to guide the terminological unification. The application of the methodological approach to the theoretical construct of practice eased the identification, collection, and analysis of the relevant state of the art about the subject. Rigor in the implementation of the methodological approach to the theoretical construct of practice ensures the results are valid basis for summarizing a definition the software engineering community could accept and share. On the other hand, a pre-conceptual schema is an efficient mechanism for extracting and synthesizing information from studies identified as relevant, since it is suitable for representing the terminology of a domain. As a feedback and improving mechanism of the methodological approach, we propose to be applied to the development of conceptual synthesis in other areas than

software engineering. Likewise, we can use the results of this conceptual synthesis in the following stages we project for reaching a unified definition of the theoretical construct of software practice.

Figure 12. Pre-conceptual schema result of the information extraction from Kualih-Beh [5]

Figure 13. Pre-conceptual schema result of the information extraction from CMMI [22]

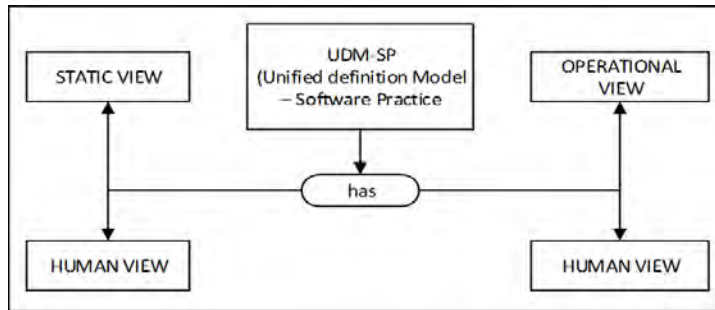


Figure 14. A Preliminary version of the definition of practice.

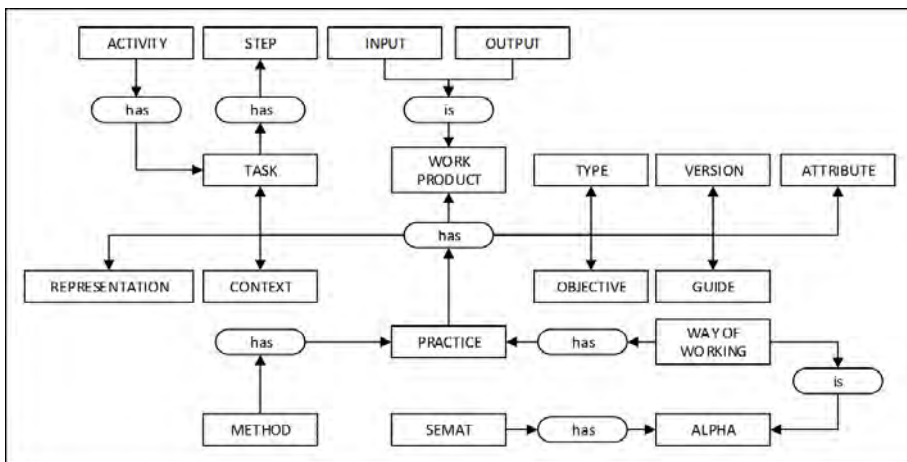


Figure 15. Static view.

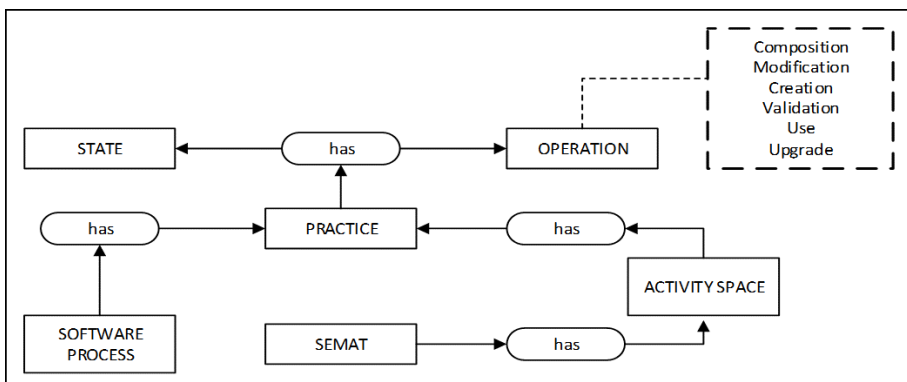


Figure 16. Operational view.

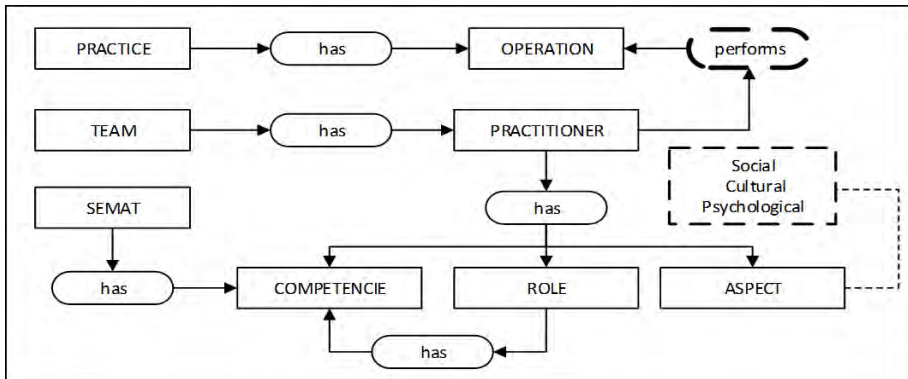


Figure 17. Human view.

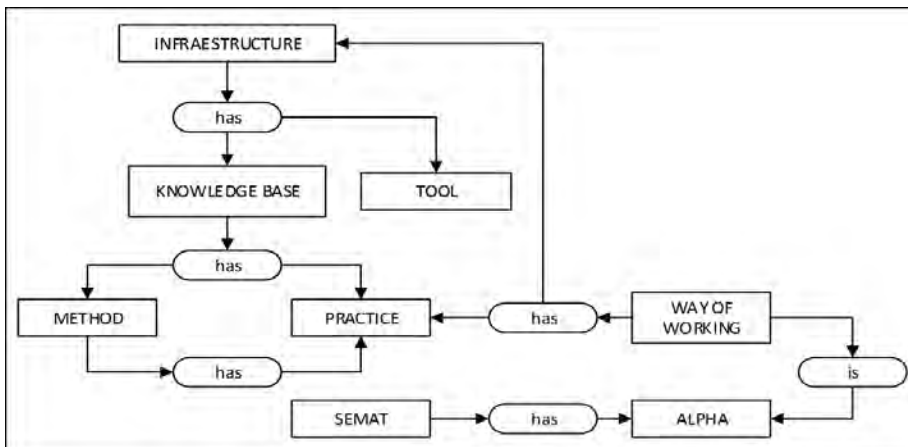


Figure 18. Management view.

## 7. References

- [1] I. Jacobson, Pan-Wei Ng, P. E. McMahon, I. Spence, and S. Lidman, "The Essence of Software Engineering: The SEMAT Kernel," *Commun. ACM*, vol. 55, no. 12, pp. 42–49, Dec. 2012.
- [2] B. C. E. Cengiz Erbas, "Modules and transactions: Building blocks for a theory of software engineering," *Sci. Comput. Program.* vol. 101, pp. 6–20, Apr. 2015.
- [3] P. Johnson and M. Ekstedt, "In search of a unified theory of software engineering," in *International Conference on Software Engineering Advances ICSEA 2007*, Cap Esterel, France, 2007, pp. 1–1.



- [4] D. Kirk and E. Tempero, "A lightweight framework for describing software practices," *J. Syst. Softw.*, vol. 85, no. 3, pp. 582–595, Mar. 2012.
- [5] Object Management Group, "Kernel and Language for Software Engineering Methods (Essence)-Version 1.1," Object Manag. Group, Feb. 2015.
- [6] C. Passos, D. S. Cruzes, T. Dyba, and M. Mendonca, "Challenges of applying ethnography to study software practices,," in *Proceedings of the 2012 ACM-IEEE International Symposium on Empirical Software Engineering & Measurement*, Ipswich, Massachusetts, 2012, p. 9.
- [7] B. Rolandsson, M. Bergquist, and J. Ljungberg, "Open Source in the Firm: Opening Up Professional Practices of Software Development," *Res. Policy*, Göteborg, Sweden, vol. 40, no. 4, pp. 576–587, May 2011.
- [8] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering, EBSE" Durham, UK, Tec. Rep, EBSE-2007-01, Jul. 2007. B. Kitchenham and P. Brereton, "A systematic review of systematic review process research in software engineering," *Inf. Softw. Technol.*, vol. 55, no. 12, pp. 2049–2075, Dec. 2013.
- [9] C. Zapata, A. Gelbukh, and F. Isaza, "Pre-conceptual schema: A conceptual-graph-like knowledge representation for requirements elicitation," *MICAI 2006 Adv. Artif. Intell.*, vol. 4293, Apizaco, Mexico, pp. 27–37, 2006.
- [10] I. Jacobson, I. Spence, P. Johnson, and M. Kajko-Mattsson, "The Essence of Software Engineering The SEMAT Approach," in *Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering*, 2012, pp. 15–19.
- [11] "Welcome - SEMAT." [Online]. Available: <http://semat.org/>. [Accessed: 19-Mar-2016].
- [12] G. W. Suter, "Review papers are important and worth writing: Review papers are important," *Environ. Toxicol. Chem.*, vol. 32, no. 9, pp. 1929–1930, Sep. 2013.
- [13] P. Lenberg, R. Feldt, and L. G. Wallgren, "Behavioral software engineering: A definition and systematic literature review," *J. Syst. Softw.*, vol. 107, pp. 15–37, Sep. 2015.
- [14] F. Selleri Silva, F. Soares, A. L. Peres, I. M. de Azevedo, A. Vasconcelos, F. K. Kamei, and S. R. de L. Meira, "Using CMMI together with agile software development: A systematic review," *Inf. Softw. Technol.*, vol. 58, pp. 20–43, Feb. 2015.
- [15] M. Zarour, A. Abran, J.-M. Desharnais, and A. Alarifi, "An investigation into the best practices for the successful design and implementation of lightweight software process assessment methods: A systematic literature review," *J. Syst. Softw.*, vol. 101, pp. 180–192, Mar. 2015.
- [16] H. Zhang and M. Ali Babar, "Systematic reviews in software engineering: An empirical investigation," *Inf. Softw. Technol.*, vol. 55, no. 7, pp. 1341–1354, Jul. 2013.
- [17] D. Heaton and J. C. Carver, "Claims about the use of software engineering practices in science: A systematic literature review," *Inf. Softw. Technol.*, vol. 67, pp. 207–219, Nov. 2015.

- [18] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software engineering domain," *J. Syst. Softw.*, vol. 80, no. 4, pp. 571–583, Apr. 2007
- [19] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering – A systematic literature review," *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, Jan. 2009.
- [20] C. P. Carnegie Mellon University, "CMMI® para Desarrollo, Versión 1.3," 2010.
- [21] SCRUM Study, A Guide to the SCRUM BODY OF KNOWLEDGE. VMEdu, Inc., 2013.
- [22] P. Kruchten, *The rational unified process : an introduction*, 3rd ed. Boston: Addison Wesley, 2003.
- [23] Rational Software Company. *Rational Unified Process: Best Practices for Software Development Teams*. Lexington, 2005.
- [24] The Eclipse Foundation, "Eclipse Process Framework Project (EPF)," 2014. [Online]. Available: <https://eclipse.org/epf/>. [Accessed: 24-Feb-2016].
- [25] Zapata-Jaramillo C.M. Barón-Salazar A. A. "Estrategia Metodológica para la Elaboración de Síntesis Conceptuales en Ingeniería de Software: una Aplicación al Caso del Constructo Teórico de Práctica," in *Proceedings of the 4th International Conference in Software Engineering Research and Innovation*, Puebla, Puebla, México 2016.
- [26] Carnegie Mellon University, "CMMI® for Development, Version 1.3," Nov 2010. (27)
- [27] ACM Digital Library. (2015, Jun), [Online]. Available: <http://dl.acm.org/>
- [28] EBSCO, <http://eds.a.ebscohost.com>.
- [29] Engineering Village. (2015, Jun), [Online]. Available: <https://www.engineeringvillage.com/>
- [30] IEEE Xplore Digital Library, (2015, Jun). [Online]. Available: <http://ieeexplore.ieee.org/Xplore/home.jsp>
- [31] ScienceDirect, (2015, Jun). [Online]. Available: <http://www.sciencedirect.com/>
- [32] Scopus, (2015, Jun). [Online]. Available: <http://www-scopus-com>.
- [33] Web of science, (2015, Jun). [Online]. Available: <http://thomsonreuters.com/en/products-services/scholarly-scientific-research/scholarly-search-and-discovery/web-of-science.html>

# Chapter # 15

## Facilitating the development of Collaborative Applications with the MVC Architectural Pattern

**Mario Anzures-García, Luz A. Sánchez-Gálvez**  
Facultad de Ciencias de la Computación,  
BUAP, Puebla, México  
{mario.anzures, sánchez.gálvez}@correo.  
buap.mx

**Miguel J. Hornos, and Patricia Paderewski-Rodríguez**  
ETSIIIT, Universidad de Granada  
Granada, Spain  
{mhornos,patricia}@ugr.es

### 1. Introduction

Collaborative application (also known as Groupware) is a computer-based system that supports groups of people engaged in a common task (or goal) and provides an interface to a shared environment [1]. This means having to promote communication, coordination, and collaboration among the group, providing network protocols and organizational structures, as well as synchronization (notification and concurrency), access control, workflow, and shared workspace mechanisms. This requires specific models to support the development process of collaborative applications, which on the one hand provide all the elements required for carrying out this process, while on the other hand it guides the developer in a natural, easy, and flexible manner through each stage of this process.

One of the most recognized architectural patterns is the Model-View-Controller (MVC). The MVC [2, 3] architectural pattern provides a template that serves as a guideline to analyze, design, and implement a software project; establishing a set of recommendations to facilitate this process of software development. In the collaborative applications domain, MVC has been used to manage the interaction among user's groups.

Consequently, in this chapter an MVC architectural pattern for developing collaborative applications is presented, in which each of the parts (Model, View, and Controller) is constituted by items that characterize this kind of applications. These items are extracted and inferred from the analysis on various works [4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, and 18] existing in the literature about this development. In order to perform this de-

velopment, the items were settled into templates which will be the guidelines to identify the requirements of collaborative applications, in a natural, clear, and flexible manner. This proposal offers a way to simplify the development of collaborative applications; providing the necessary flexibility and responsiveness to adjust to the changing needs within the group.

Furthermore, this document is an extended version of the chapter presented in [19], in which the state of the art, corresponding to the MVC architectural pattern, and the collaborative applications, has been reduced. So as also, a template and several figures have been added in order to further explain the importance of the Templates (which are derived from the MVC proposal) in the development of such applications.

The chapter is organized as follows. The MVC architectural pattern is briefly discussed Section 2. The state of art about the development of collaborative applications is outlined in Section 3. The MVC architectural pattern and templates to guide this development are shown in Section 4. A case study is unfolded in Section 5. Finally, conclusions and future works are given in Section 6.

## 2. MVC architectural pattern

The concept of design patterns was settled in the seventies by Alexander and his colleagues [20, 21, and 22] in order to preserve the fundamentals concepts of architecture within the new modernist tradition. Every pattern describes a recurring problem, its context, the forces that are at play in the situation, and a solution to the problem. The feature that solves the problem is written in a generic but concrete form, so it can be designed in an unlimited number of ways while still being readily identifiable [21]. So, the design patterns provide: techniques proven that reflect the experience and insights the developers; solutions adapted for suiting our own needs; and expressiveness required to present complex solutions in a simple way.

An architectural pattern captures the essence of a successful solution to commonly occurring problems in software design. Thus, a pattern can be seen as a clear and generic set of instructions, ensuring to use a solution that has been proven in countless software design problems with excellent results, allowing customize the pattern to solve specific problems. The importance in the architectural pattern approach is its potential to bridge the gap between high-level requirements and design. So the architectural patterns may be used during requirements elicitation to determinate the necessary items to develop software and how these are applied to the pattern [22]. They can also advise the design process and aid designers with the development of prototypes.

MVC is an architectural pattern that encourages an improved application organization through a separation of concerns. This pattern was originally designed by Trygve Reenskaug during his time working on Smalltalk-80, where it was initially called Model-View-Controller-Editor [2]. MVC is created to reduce the cost and improve the quality of software in the object-oriented paradigm. MVC improves modularity by encapsulating volatile implementation details behind stable interfaces that reduce the effort required to understand and maintain existing software.

The Model represents the knowledge domain of the application; characterizing unique forms of data in an application. When a model changes (it is updated or modified), it will typically notify its views (observers) that a change has occurred, so that they may react suitably. View is a (visual) representation of its model. It would ordinarily highlight certain attributes in the model and suppress others. A view typically has associated a model and is notified when the model (or a part of it) changes, allowing the view to update itself accordingly. All these notifications must be in the model terminology. Users are able to interact with views, and this includes the capacity to access and modify the model. Controller is the link between a user and the application. It provides the user with input by arranging for relevant views to present themselves in suitable places on the screen. It receives user output, translates it into the appropriate messages and passes these messages to one or more views.

Some examples of using MVC in collaborative applications, are: Groupkit [4, 5] —tool-kit— and CLOCK [6] — architectural model— using MVC to support real-time, distance collaborative work; AORTA (Action-oriented decoupled architecture for Coordination and Awareness) [7], which follow replicated or MVC hybrid variants for integrating new components; and two applications to solve specific problems, such as [8] and [9] that are based on MVC for managing content on the cloud, and creating collaborative framework of the test set, correspondingly.

In summary, these jobs use the MVC for supporting the users' interaction based on data structures or documents. However, they not use MVC to develop a collaborative application, let alone providing a set of items for each component of the model, as in this chapter. In addition, some templates are proposed to simplify this development.

### **3. State of the art about collaborative applications**

Nowadays, the collaborative work is a tool to integrate the work group -whose members are not physically in the same place- and require creating (conceptual) products or ser-

vices. The use of such tools has increased, as they facilitate the communication, coordination, and collaboration of researchers and/or stakeholders. However, it is necessary to model the process of these tools' development by means of formalisms, which substantiate the communication, both synchronous and asynchronous, the interaction among members, and between the latter and the application, the integration among its components and the whole process development. Besides, these formalisms must be presented in a natural, clear manner, being that they function as guidelines to simplify this process.

Many fields of science make use of formalisms, such as theories, rules, patterns, methodologies, and methods to precisely predict outcomes given certain inputs, and to explain some phenomena of the universe [23]. In the engineering software [24] as well as in the collaborative domain, AMENITIES (a methodology for analysis and design of cooperative systems) [10], and [11] to support the process of collaborative applications development; CIAM (Collaborative Interactive Applications Methodology) [12], and TOUCHE (Task-Oriented and User Centered process model for developing interfaces for Human-Computer-Human Environments) [13], to manage the interaction among the users; some samples have been presented. Three of these jobs by using notations based on UML, and AMENITIES by an extension named COMO-UML. In addition, works using notations of models [14, 15, 16, and 17], and meta-models [18] have also been exposed. In which different concepts, and ideas have been used to the design and carry out construction processes of new collaborative applications. Examples of concepts are: group, role, actor, task (sequential, parallel, additive concurrent, and fully concurrent), activity, resource, session management policy, session, notification, group awareness, group memory, concurrency, shared user interface, stage, task precedence, division of labor, coordination at activity-level and at object-level, information view, participant view, context view.

In conclusion, the aforementioned theories, models, and methodologies provide a set of essentials concepts for developing a collaborative application. Some of these concepts are even considered in the proposal for this article. Moreover, these jobs are sustained with notations by using models, UML, or variants of it. However, neither of the jobs mentioned above supplies a structure to assemble the concepts (which would facilitate the building of a collaborative application), nor it gives any guidelines to rule and simplify the work of software developers. On the other hand, these formalisms cannot be considered formal, but informal. In this chapter, the concepts are grouped in three parts (Model-View-Controller) so reducing the developing time and cost of a collaborative application. The templates are derived from these groupings for guiding the developers in such a way that the building of this kind of applications is simplified. What is more, the Model is based on an ontology that specifies the group organizatio-

nal structure (which defines the policy based on a division of labor, taking into account the roles the users may play) allowing the evolution, reuse and adaptation of collaborative applications, because ontology can be adjusted by adding concepts, relations, and instances. The ontology is a notation that supplies a (well defined), common and shared vocabulary, which provides a set of terms, relations, and rules for describing this domain in a formal way.

## 4. MVC based-development of collaborative applications

In this chapter, the MVC design pattern has been chosen for four reasons:

1. It allows separating the development of collaborative applications in three components, where each one plays its own separate role, making the application to work appropriately. This separation facilitates the reuse of the model by several user interfaces, and it organizes the development of an application in a structured way.
2. It is used in the object-oriented analysis and design; therefore, it can be applied in the requirement analysis and design of software.
3. It is ideal for building collaborative applications, because it reduces the time, effort and cost of this process.
4. It enables to reuse the pattern components, since these are designed independently.

The items of each component of the MVC architectural pattern are derived from the studies performed in [4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, and 18], as well as from the ideas and concepts analyzed in the previous sections:

- The model of objects, coordination, and user interface must be specified to develop collaborative applications.
- Events allow triggering tasks.
- Tasks, activities and actions are the medium to achieve a common goal.
- The task precedence must be established in the collaborative applications.
- A task flow must be created; therefore, a workflow is recommended, since it coordinates the execution of multiple tasks or operations along the development process.

For this reason, the tasks must be placed in a Template, then; the dependencies between these must be established by a task flow; finally, a coordination mechanism must be created and related to each task.

- Tasks can be executed into stages (when it comes to developing complex collaborative applications). These must be monitored to know the state of each task.
- The task structure requires to establish an organizational structure, which allows to define a division of labor in accordance with the changing needs of the group and the application.
- Two coordination levels must be considered: activity-level and object-level.
- Three access types to the objects or resources are considered: parallel, additive concurrent, and fully concurrent.
- A notification mechanism must be provided to update the objects being accessed, and a concurrency one, to avoid the simultaneous modification of the same object.
- The context of every activity is essential to the work environment for it is necessary to take into account the activities' internal and external resources.
- The user interfaces can be determined by identifying the resources and the users' roles involved.
- In collaborative applications, three kinds of user interfaces such as information view —showing objects—; participant view —group awareness—; and context view —group memory—; can be considered.

In the next subsections, the items for the Model, View, and the Controller will be defined.

## 4.1 Model items

Model of the MVC architectural pattern contains the data required for developing collaborative applications in a suitable way. Therefore, in this chapter, the Model will specify the items of the group organizational structure, which is founded on the ontology modeling the session management policies [25, and 26]. This ontology coordinates and adjusts the collaborative applications in accordance to its changing needs. This structure defines the policy based division-labor considering the roles the users can play.

This ontology that establishes the Group Organizational Structure (see the yellow rectangles in in Figure 1) is governed by a specific policy which determines how the group



is organized. This structure is made up of users. Policy; it defines a configuration of the group organizational structure in accordance with each role established. Users; they can be people, either individuals or in groups, although they may also refer to systems playing one or more of the established roles. Role; it is responsible for the tasks that users can perform and it provides a set of access rights —related to its status— on the shared resources used to carry out the activities. Status; it describes the role hierarchy, i.e. it finds the role priority in the group. Task; it is made up of one or more activities, allowing users to achieve a given goal in a certain period of time. Activities; these are actions that allow a role to execute a set of operations. Shared Resource; it represents the resources used to carry out the activities.

In this chapter, the group organizational structure is extended (see the green rectangles in Figure 1) to supply coordination on activity-level and object-level. In the former, the Events, and the task sequence are added; one for triggering each task, and another one to manage them. Furthermore, the inclusion of the Stage facilitates the tasks' sequence, since one set of tasks is defined by it, as well as the order that they may have. In the latter, three types of tasks are aggregated: The Sequential-Task to access to different objects; the Parallel-Task to manage the same parallel object; and the Concurrent-Task to use a unique object concurrently. Therefore, a notification mechanism should be provided for the three tasks, and a concurrency one for managing the third task.

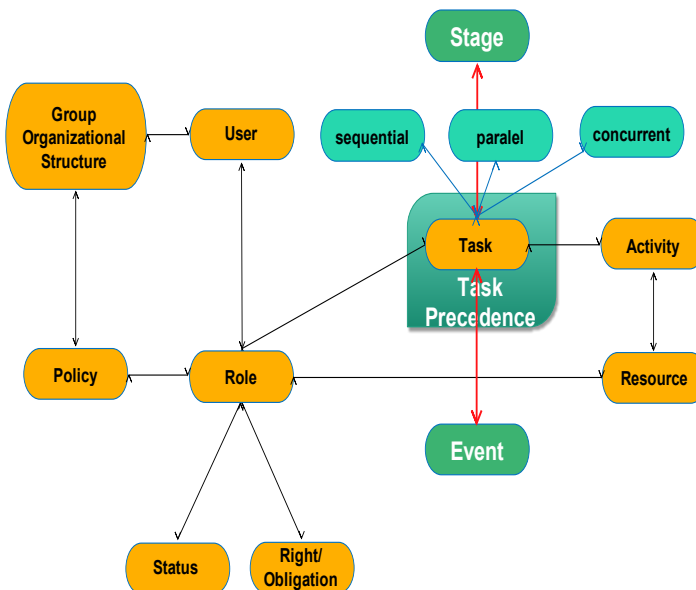


Figure 1. Model items to build a collaborative application.

The group organizational structure outlines a particular style for carrying out group work in accordance with every item that composes it. However, collaborative application requires supporting different styles. For this reason, the group organizational structure extended is an ontological model, which supplies the flexibility necessary in order to adapt the group to new organizational structures by simply adding new instances to the ontology, for evolving this structure changing, adding, or removing both concepts and relationships between themselves.

In addition, the Task Precedence is characterized by a workflow. Consequently, this workflow will represent any set of tasks —along with their order of execution— performed by different roles for achieving a common goal. Stage concept has been added to simplify the users' access to shared workspace, considering that the users only are admitted on certain collaboration moments, when they must carry out a task. Both the phase and the workflow are ideal for controlling the activity-level coordination.

## 4.2 View items

Views are user interfaces that show the resources and the interaction among users, and between them and the application. The session is provided through the views, allowing the users to interact with the application. The controller manages the information of the model to generate the appropriate view; thus, several views can be created by a same model. In the case of collaborative applications, three views are considered (see Figure 2): The Information View (IV), the Participant View (PV), and the Context View (CV).

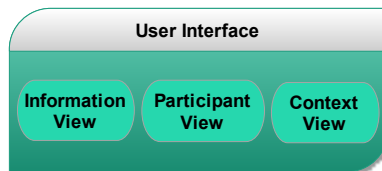


Figure 2. View items to develop a collaborative application.

The Information View allows seeing all the information that helps the user to interact with the collaborative application; for this reason, it displays all the tasks performed by one or several roles over resources and where other users have no involvement. This view shows modifications related to sequential-task.

The Participant View allows each user to be aware of what other's users are doing, and therefore they know what happens in the shared workspace, since the group is (generally) distributed in different geographical locations, working through the computer. Consequently,

this view provides group awareness through views showing the notifications of the distinct changes produced by other group users, who are participating; hence, the name of “participant view”. Collaborative application supplies widgets to show other users all that is happening in the application. Every collaborative activity triggers a notification, which sends messages to update the views, concerning the modification of the resource, promoting greater cooperation and interaction among users in different collaborative tasks. To do this, concurrency is used, for it helps to manage the permissions assigned to users, in order to make use of shared resources; thus ensuring the mutually exclusive use thereof. The modifications carried out by parallel-tasks and concurrent-tasks are displayed in this view.

The Context View represents the common workspace where all information of shared resources is shown; this view is named memory or history group. It displays the cooperative activities of the group, this was created to provide understanding and reasoning about the collaborative process unfold, towards making an accurate monitoring. Furthermore, it is related to the notification; allowing to store, collect, and distribute information by supporting the group knowledge representation, in a structural, dynamical manner. Group memory is generated to store the information of the shared resources used and the activities performed by each role involved in the collaborative application. This view exhibits the changes implemented by parallel-task and concurrent-task.

User interface can display three, two, or one view, in accordance with the executed task, since a task can determine the type of coordination (activity-level or object-level) to execute.

### 4.3 Controller items

The Controller manages and updates the views appropriately in accordance with the shared resources modification of the model, which results from the users’ interaction. In the collaborative applications, the interaction is carried out during the session, and the user interfaces (that compose the same) display the changes of the model. The controller operates the notification mechanism, updating the information view for the case of a sequential-task, as well as the participant view, and the context view, when a parallel-task or concurrent-task is produced. In the latter case, the concurrency mechanism is also activated. This mechanism avoids conflicts when a resource is being used by several participants, which is based on the organizational structure extended (see Figure 3).

The Session establishes a shared workspace, where a group of people sharing a common interest will work, it supplies a session management mechanism to control and handle sessions through a user interface, by which users establish a connection; that is, for users to join, leave, invite someone to, and exclude somebody from a session. Generally,

this mechanism specifies how the group will be organized. Nowadays, the session management is uncoupled to the group organizational structure, since the session is implemented with the methods or functions provided by the framework or the programming language used, and the organizational structure is performed by the developer.

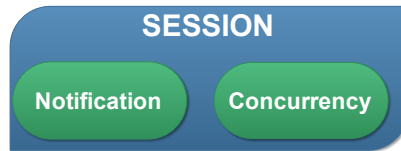


Figure 3. Controller items to create a collaborative application.

A notification occurs when one or all the shared resources are modified in order to keep the views updated. The notification must provide information regarding the events in a session related to a user, a subset of users, or an entire group. This is essential to coordinate the users' activities, and it provides the group awareness, i.e. a user knows what others are doing in a shared workspace through events or widgets presented in the user interface. In addition, each event that occurs can be stored and displayed in the context view, generating group memory.

The concurrency determines in what way the participants in a session contribute, providing dynamically-generated temporary access and manipulation permissions for collaborating users, in order to reduce competition conditions and to guarantee mutually exclusive resource usage. The permissions are granted to users depending on the roles that they play; in such a way, these permissions specify which users can send, receive or manipulate shared data at a given stage.

Notification and concurrency, control the interaction occurred during the session and adapt the corresponding views in accordance with the type of the executed task. Furthermore, these mechanisms must be implemented in each function or method that represents a task. The adaptation is accomplished through the notification, and controlled by concurrency, in order to adjust views in agreement with the performed interaction. As mentioned above, the adaptation can be applied if new instances are created on the ontology, or even when new concepts and/or relations are also added.

Once each of the items has been allocated in the three layers, the MVC architectural pattern (see Figure 4) to develop collaborative applications is ready for usage. However, it can be difficult for inexperienced people in this domain, to create software with the MVC pattern shown in the Figure 4. Thus, three Templates (one for each MVC component) are derived of the MVC in order to simplify and facilitate the development of the collaborative applica-

tions. Since the developer must only specify and place the elements (ontology instances) in each Template column (item). The columns from each Template will be the items of the Model, the View, and the Controller; except for the items called Group Structure Organizational, Policy, and User, which are not shown. Consequently, the Template related to the Model will have 12 columns. The Template of the View will have four items, and the Template corresponding to the Controller will have three items. These items established can be used for both the analysis and the design of an application; since all the necessities to build up a collaborative application are represented in these Templates.

Once the Template Model is filled (i.e., the requirements analysis of the Model is completed), the following can be generated:

- The ontological model of the organizational structure presents the roles —their status and rights/obligations— that establish the tasks to be performed, their activities and the resources used.
- The task precedence workflow provides the stages order, and of their tasks in them, as well as the events which activate each task.
- The access control establishes the roles participating in each stage.
- The partial data model —relational or not—shows the data, and the relations that exist among them.

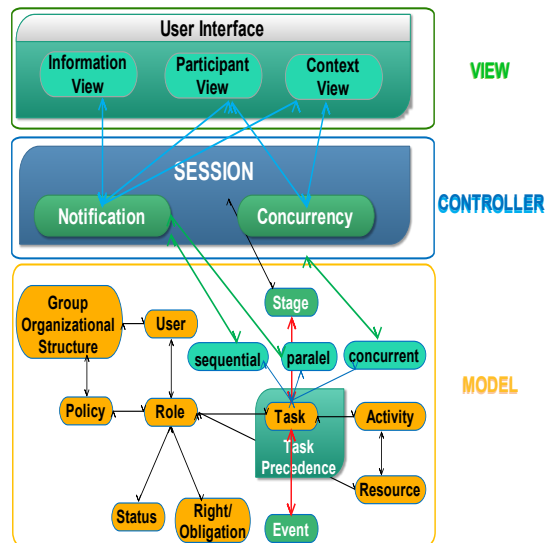


Figure 4. MVC design patten to develop a collaborative application.

Once the Controller Template has been completed, it is possible to deduce:

- The coordination workflow considers the resources used in the concurrent task and the role being carried out, as well as, the managed mechanism, both the notification and the concurrency.
- The collaboration workflow displays the different roles that work together on the execution of a task.
- Once the View Template has been created, the following can be generated:
- The interaction workflow shows the user interfaces and the views contained in them, as well as the roles participating in each task shown on the user interfaces.
- The group awareness workflow exhibits the Participant's View, as well as the modified resources for tasks involved in this view.
- The group memory workflow presents the Context View, as well as the resources modified by the tasks involved in this view.

In summary, the inclusion of the three types of tasks simplifies the management of the Controller, as they help to shorten notification and concurrency processes, as well as to define that view will be shown in the different user interfaces that present the session. So, the developer will have a suitable design to its application.

Finally, the implementation of the collaborative application can be achieved, by codifying the tasks and user interface allocated on the Templates. Furthermore, the modulation is made through MVC architectural pattern presented.

#### **4.4 Using the MVC architectural pattern proposed to develop a collaborative application**

A study case is presented to demonstrate the operability of the MVC architectural pattern using the Templates. The study case consists in the development of a collaborative application for managing Departmental Test (DET) of the Facultad de Ciencias de la Computación de la Universidad Autónoma de Puebla. The DET homogenizes the teaching of a subject, i.e. it guarantees that all teachers encompass the same percentage of the academic program. For this reason, it requires a shared workspace that allows professors to manage and apply a DET. Several roles are considered in DET: The Manager (Mg) who configures the application (CA) and has status equal to 1, so, he/she registers the users, who play the other four roles, the knowledge areas, and the subjects that are a part of them. The Area Coordinator (AC) with status 2, who manages the test (MT), so, he/she registers the TC and schedules the professors' meetings, related with the same subject. The Test

Coordinator (TC) with status 3, who organizes the test (OT), so, he/she put in order the completion of each test, requesting and agreeing on the number of tests to be applied, as well as on the dates and the number of questions which will be included; then he/she will post the test and the classroom, where each Professor will apply it. The Professor (P) with status 4, who generates the test (GT), thus, he/she will propose and vote the date when the test will be performed, so as the number of questions contained in the exam.

**Table 1. Template of the test elaborating stage.**

MODEL								VIEW CONTROLLER									
Stage	Role	St	R/O	Event	TASK	ACTIVITY	RESOURCE	Pre	U	E	C	U	U	U	U	U	U
TC, P	TC, P	3, 4	OT, to define GT test date	to access to DET	Au	getting into data	Text box	1	Ö	Ö	X	Ö	Ö	Ö	Not shared	Ö	X
						sending data	Accept Button										
					PD	getting into date	Test UI	2	Ö	Ö	X	Ö	Ö	Ö	Shared	Ö	X
					SD	posting date											
						choosing date		6	Ö	Ö	X	Ö	Ö	Ö	Not shared	Ö	X
						loading date											
P	P	4	GT	to access to DET	Au	getting into data	Text box	3	Ö	Ö	X	Ö	Ö	Ö	Not shared	Ö	X
						sending data	Accept Button										
					CP	choosing data	Coordinator UI	4	Ö	X	X	Ö	X	X	Not shared	X	X
						showing data	Accept Button										
					CD	choosing date	Scheduling UI	5	Ö	Ö	Ö	Ö	Ö	Ö	Shared	Ö	Ö
						vote date	Text box										
TE	TC, P	3, 4	OT, GT	to select DET questions	PNQ	posting data		7	Ö	X	X	Ö	X	X	Not shared	X	X
						posting data											
					SNQ	choosing number	Test UI	8	Ö	X	X	Ö	X	X	Not shared	X	X
						loading number											
					CP	visualizing question		9	Ö	X	X	Ö	X	X	Not shared	X	X
						choosing question											
					PQ	loading question		10	Ö	Ö	Ö	Ö	Ö	Ö	Shared	Ö	Ö
						choosing file	Test UI										
					LP	Subir file		11	Ö	X	X	Ö	X	X	Not shared	X	X
						choosing data	Coordinator UI										
					CP	showing data	Accept Button	12	Ö	X	X	Ö	X	X	Not shared	X	X
						choosing file	Test UI										
					DE	downloading file		13	Ö	X	X	Ö	X	X	Not shared	X	X
						choosing question											
P	P	4	GT	to define DET	CQ	vote by question	Text box	13	Ö	Ö	Ö	Ö	Ö	Ö	Shared	Ö	Ö
						sending vote	Accept Button										
					PN	writing notice	Text box	0	Ö	X	X	Ö	X	X	Not shared	X	X
						posting notice	Accept Button										
					PM	writing message	Text box	0	Ö	X	X	Ö	X	X	Not shared	X	X
						posting message	Accept Button										

Table 2. Template of the test elaborating stage.

MODEL										VIEW				CONTROLLER			
Stage	Role	St	R/O	Event	TASK	ACTIVITY	RESOURCE	Pre	St	Pt	Ct	IV	PV	CV	SS	Nt	Cc
Tf	Mg	1	CA	access to DET	Au	CRUD data	Text box	1	Ö	X	X	Ö	X	X	Unshared	Ö	X
				Accept Button													
				to manage AC	CRUD AC		Form	2	Ö	X	X	Ö	X	X	Unshared	Ö	X
				Accept Button													
				to manage area	CRUD Area		Form	3	Ö	X	X	Ö	X	X	Unshared	Ö	X
				Accept Button													
to manage Subject	CRUD Subject	Form	4	Ö	X	X	Ö	X	X	Unshared	Ö	X					
Accept Button																	
to manage P	CRUD P	Form	5	Ö	X	X	Ö	X	X	Shared	Ö	X					
Accept Button																	

Table 2. Template of the test preparation stage.

MODEL											VIEW				CONTROLLER		
Stage	Role	St	R/O	Event	TASK	ACTIVITY	RESOURCE	Pre	St	Pt	Ct	IV	PV	CV	SS	Nt	Cc
TP	AC	3	MT	to access to application	Au	getting data	Text box	1	Ö	Ö	X	Ö	Ö	Ö	Not shared	Ö	X
						sending data	Accept Button										
				to register	CRUD TC	CRUD TC	form	3	Ö	Ö	X	Ö	Ö	Ö	Not shared	Ö	X
							Accept Button										
				to scheduling	PMD	writing date	Scheduling UI	2	Ö	Ö	X	X	Ö	Ö	Not shared	Ö	X
						posting date						X	X	X			
	SD	choosing date		Ö	Ö	X	Ö	Ö	Ö	Not shared	Ö	Ö					
		loading date		Scheduling													

Table 3. Template of the test results stage.

MODEL								VIEW				CONTROLLER						
Stage	Role	St	R/O	Event	TASK	ACTIVITY	RESOURCE	Pre	ST	PT	CT	IV	PV	CV	Ss	Nt	Cc	
TR	TC	3	OT	To create test	LDET	loading file posting file	Test UI	1	Ö	Ö	X	Ö	Ö	Ö	Shared	Ö	X	
					PoC	writing classroom posting classroom	Text box Acept Button	2	Ö	Ö	X	Ö	Ö	Ö	Shared	Ö	X	
						P	4	GT	To posting scores	LoS	loading file posting file	Test UI	3	Ö	Ö	X	Ö	Ö
	S, P, TC, AC	5, 4, 3, 2	VS, GT, OT, CA		DoS	choosing file downloading file	Test UI	4		Ö	Ö	X	Ö	Ö	Ö	Shared	Ö	X

This table continues on the following page→



MODEL								VIEW							CONTROLLER		
Stage	Role	St	R/O	Event	TASK	ACTIVITY	RESOURCE	Pre	ST	PT	CT	IV	PV	CV	Ss	Nt	Cc
TR	P, TC, AC	4, 3, 2	GT, OT, CA	To generate reports	GR	creating report	File	5	Ö	Ö	X	Ö	Ö	X	Un-shared	Ö	X
						loading report											
					LS	loading file	Test UI	6	Ö	Ö	X	Ö	Ö	Ö	Shared	Ö	X
						posting file											
	S, P, TC, AC	5, 4, 3, 2	VS, GT, OT, CA	To Chat	PM	writing message	Text box	0	Ö	Ö	X	Ö	Ö	Ö	Shared	Ö	X
						posting message											
	P, TC, AC	4, 3, 2	GT, OT, CA	To public notice	PN	writing notice	Text box	0	Ö	Ö	X	Ö	Ö	Ö	Shared	Ö	X
						posting notice											
	S, P, TC, AC	5, 4, 3, 2	VS, GT, OT, CA	To Schedule test	ST	Scheduling	Scheduling UI	7	Ö	X	X	Ö	X	X	Shared	X	X
							Scheduling										

The Students (S) with status 5, who views scores (VS) of the test, so, he/she will look up the information about the date and classroom, where the test will be carried out, as well as to find the grades obtained on each subject. In general, the five roles must register to join at the session.

The collaborative application for managing the DET has four stages: Test Configuration (Tf), Test Preparation (TP), Test Elaborating (TE), and Test Results (TR). Which will be explained later.

The requirements analysis can be carried out by using techniques of classic or agile methodologies. It depends on the developer abilities. Once the developer has got the requirements, he/she allocates them on the Templates, which are generated in this chapter. For reasons of space, four Templates are presented (see Table 1, 2, 3, and 4), one by each DET stage. Each Template contains the columns related to the three components of the MVC architectural pattern; with the exceptions mentioned above.

The organizational structure ontology of DET is derived from the Model Template. The figure 5 shows this structure partially, in which the five roles of the DET with its status (St) and Right/Obligations (OR) are presented, but only two of these roles are active, because they correspond to the stage of TE, which is shown in this figure. Therefore, for these roles (TC and P) their tasks together with their types (Sequential –Sq, Parallel –Pr, Concurrent –Cn), and precedences are displayed. The events, activities and resources for the tasks of the roles TC and P for reasons of space are not exhibited.

The Stage TE proposals two roles:

- The role TC carries out the tasks of: Authentication (Au) him/herself, Proposing Date (PD), Setting Date (SD), Proposing Number of Questions (PNQ), Setting Number of Questions (SNQ), and Posting Questions (PQ);
- The role P executes the tasks: Au him/herself, Consulting Proposals (CP), Choosing Date (CD), Loading Proposal of questions (LP), Downloading Exercises of the test (DE), Choosing Questions of the test (CQ), Posting Notice (PN), and Posting Message (PM).

Moreover, all tasks are of type sequential (Sq), and some, they can also be parallel (Pr) or Concurrent (Cn). On the other hand, the task with precedence equal to zero indicates that can be carried out at any moment; once the user has accessed to DET.

Furthermore, Templates help us to build the workflow of task precedence (see Figure 6) for a user playing the role of TC. In this case, the user or users playing this role can access to three stages: TP, TE, and TR. In the first stage, the TC performs the Tasks of: Authenticate (Au) him/herself, and the functions of persistent storage, i.e., create, read, update, and delete (CRUD) for Professor. In the second one, the TC executes the Tasks of: Au him/herself, PD, SD, PNQ, Setting Number of Questions SNQ, and PQ.

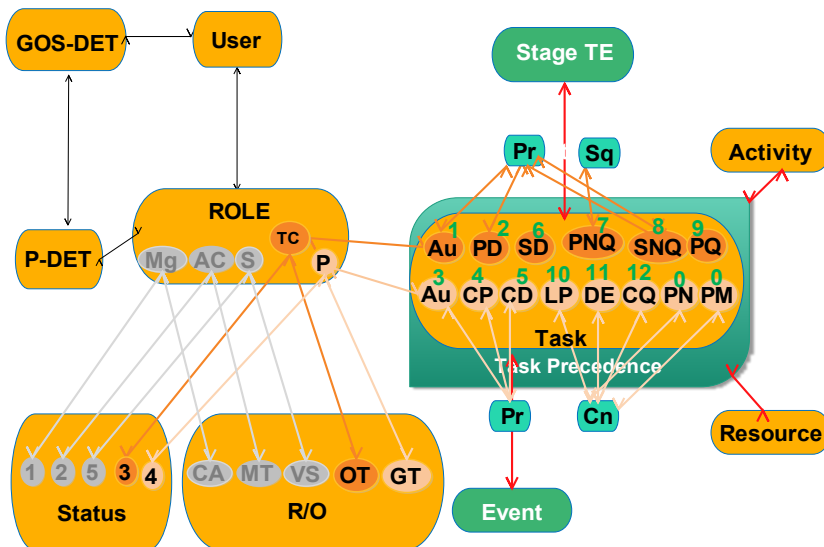


Figure 5. Organizational structure ontological model of Stage TE.

In the third one, the TC performs the following tasks: Au him/herself, Downloading Scores (DoS), Generating Reports (GR), Loading Statistics (LS), PM, PN, Scheduling Test (ST), Loading DET (LDET), and Posting classroom (PoC). For the second stage named

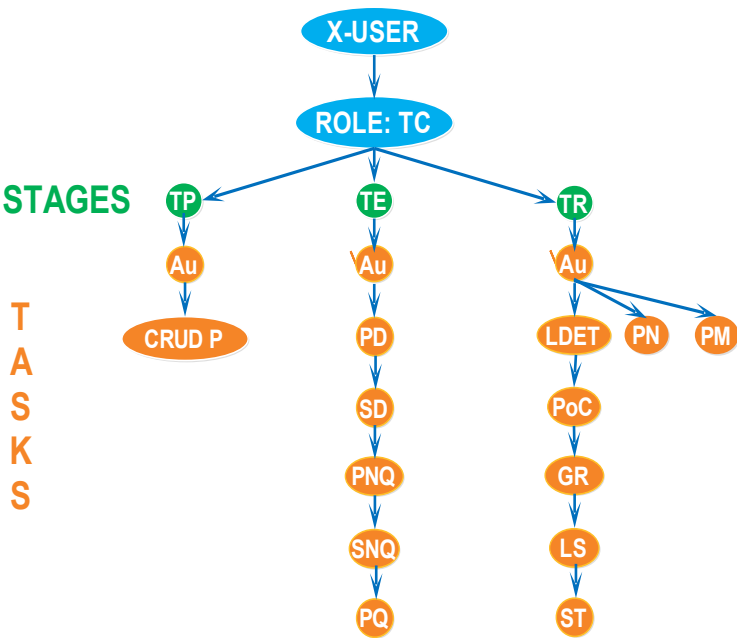


Figure 6. Task workflow of the Test Coordinator.

Test Elaborating, a User Interface is created; it allows TC to consult the proposed questions for test, select the more voted questions, and with these questions to create the test to be applied; with this a new user interface is presented, where the TC can print the test in pdf and send it to the participant professors.

For reasons of space, the tasks' precedence and access control into the Figure 7 are shown. Therefore, the four stages with precedence, and tasks that contain these, also, its respective precedences are displayed.

- The role Mg joins to the Stage Tf, executing the tasks of: Authenticate (Au) him/herself, CRUD for AC, area, subject, and AC.
- The roles Mg and TC enter to the Stage TP; the former performing the tasks of Au, CRUD TC, Proposing Meeting Date (PMD), and SD; the latter executing the tasks of Au, and CRUD P.

- The stage TE has already been described in the beginning of this section.
- Finally, four roles (AC, TC, P, and S) participate in the Stag TR. The role AC implements the tasks of DoS, GR, LS, PM, PN, and ST. The role TC in addition to performing the same tasks that the role AC, he/she carries out the tasks of LDET, and PoC. The role P effects the tasks of: Loading Scores (LoS), DoS, GR, LS, PM, PN, and ST. The role S carries out the task of DoS, PM, ST.

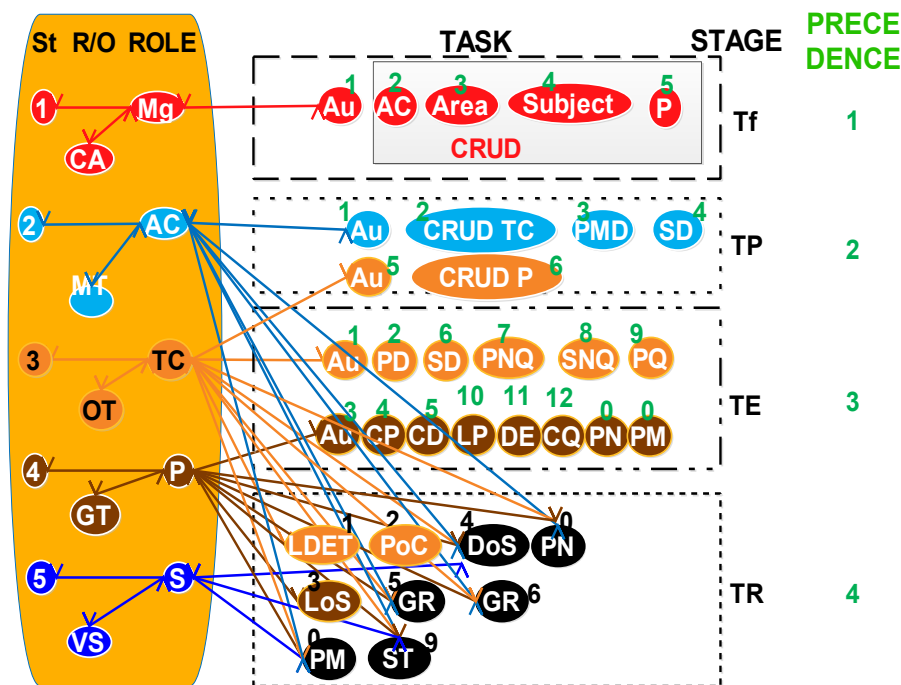


Figure 7. Precedence and access control of the DET.

Once the Model Template has been defined, then the Controller Template can be established in a simple, clear way. First, for each stage a session is implemented. Second, for each task Sequential and Parallel a notification mechanism is carried out. Third, for each task concurrent a concurrency mechanism is performed.

The workflow of collaboration and Coordination (see figure 8) is deduced from the Model and Controller Templates, and corresponds to the task LP of the Stage TE, where the user playing the role P can load its proposals of departmental test questions. Therefore, it must carry out both coordination as collaboration mechanisms. In the former, each P

sends its questions through of their user interface using the notification mechanism. These questions are stored on database in different register by concurrency mechanism, avoiding of this manner data inconsistency. In the latter, each P collaborates to create the departmental test, sending its proposals exercises, once they have been stored and displayed; the role P votes for your preference. So, the test is completed.

Once both Model Template and Controller Template have been completed, then View Template can be accomplished in a simple, clear manner. First, for each Task Sequential an Information View is presented. Second, for each Task Parallel a Participant View is shown. Third, for each task concurrent a Context View is displayed.

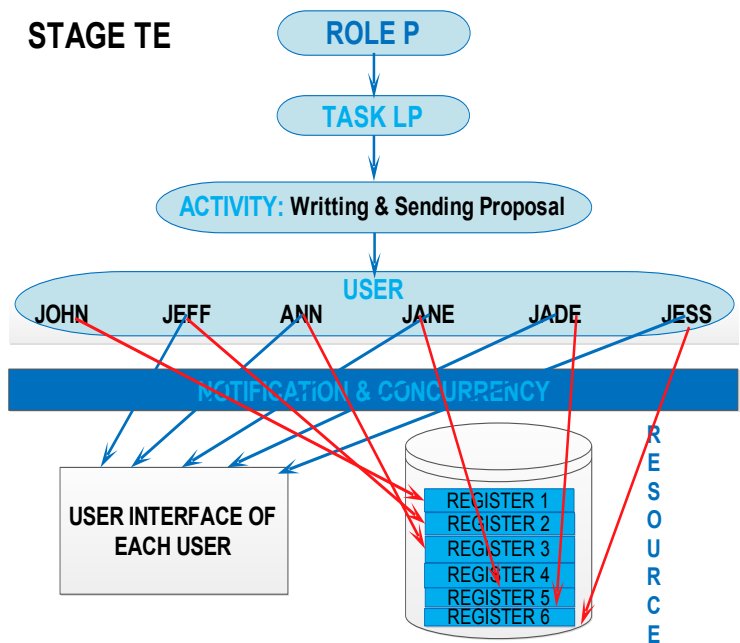


Figure 8. Workflow of collaboration and coordination of stage TE.

The Figure 8 is a good example of the workflow of interaction among the users, in this case, playing the same role (P). However, according to Table 1, the role TC can carry out the PQ (task) of departmental test. So, it will have the interaction of different users playing two roles (TC and P).

The group awareness can be watched in the figure 9, in the part right labeled as PV, where the Professors' chat is presented. Here each user with role P can send a message with respect to the departmental test that is being created.

On the other hand, the group memory can be examined in the figure 9, in the part central labeled as CV, where the areas belonging to each teacher are displayed.

This information is useful for the TC, allowing him/her to group each user with role P in a certain area.

The developer can use a framework based on MVC for the implementation of collaborative applications, reducing the effort required to do so. Therefore, the time, effort, and cost of developing this sort of applications is diminished.

For example, the user interface displayed in the Figure 9 (Spanish language is used, because the application named GEDEX is developing in Puebla, México) is created from the items of Table 1, which are related to the Mg. role. The table items —Event, Information view (IV), Participant View (PV), and Context View (CV) — are highlighted in the figure 9, in order to appreciate how these are used to create a part of the user interface.

Another example is presented in Figure 9, in which the Manager's tasks are exhibited like a menu on the left side, on the center, several postings both of the Test Coordinator (TC) and of the Professor (P) are shown, as well as some statistics. The right side displays the users who participate (active users) and those who do not.

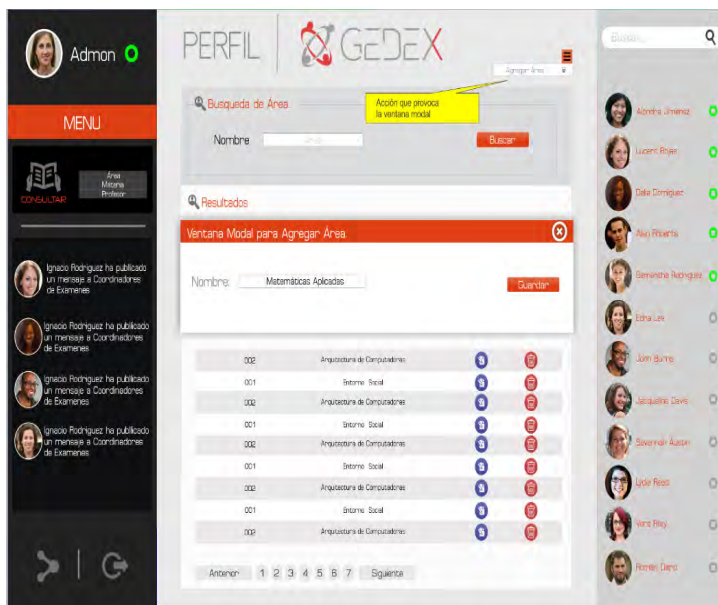


Figure 9. MVC design patten to develop collaborative applications.

## 5. Conclusions and future work

The proposal presented in this chapter allows customizing the MVC architectural pattern in order to develop a collaborative application. In this manner, a solution that has been attested in countless software problems given the same good results has been used. This MVC architectural pattern simplifies the development of such applications, supplying a set of items to support this process group. These items provide both individual and collective aspects, facilitating the appropriate flexibility and responsiveness during the evolution of the collaborative work. Finally, the templates generated from MVC architectural pattern, offer guidelines for the analysis, design, and development of groupware applications. For example, considering the items allocated in the templates, it is possible to create workflows to simplify and clarify the task precedence or the interaction of the users with the application. The future work will be focused on creating a methodological approach based on MVC design-pattern for developing groupware.

## 6. References

- [1] C.A. Ellis, S.J. Gibbs, and G.L. Rein, "Groupware: Some Issues and Experiences," *Communications of the ACM*, Vol. 34-1, Jan. 1991, pp. 39-58, doi: 10.1145/99977.99987.
- [2] A. Goldberg, *Smalltalk-80: The Interactive Programming Environment*. Addison-Wesley, 1984.
- [3] O. Addy, *Learning JavaScript Architectural patterns*. O'Reilly, 2015.
- [4] M. Roseman, and Greenberg, S. "Building real-time groupware with GroupKit. A groupware Toolkit," *ACM Trans. Comp.-Hum. -Interact.*, vol. 3, Mar. 1996, pp. 66-106, doi: 10.1145/226159.226162.
- [5] M. Roseman, and S. Greenberg, "Registration for Real Time Groupware," *Research Report 94/533/02*, Department of Computer Science, University of Calgary, Alberta, Canada, 1994.
- [6] T. Graham, C.A. Morton, and T. Urnes, "ClockWorks: Visual Programming of Component-Based Software Architectures," *Journal of Visual Languages and Computing*, vol. 7-2, June 1996, pp. 175-196, doi:10.1006/jvlc.1996.0010.
- [7] P. Orozco, J.I. Asensio, P. García, Y.A. Dimitriadis, and C. A. Pairot, "Decoupled Architecture for Action-Oriented Coordination and Awareness Management in CSCL/W Frameworks," *Lecture Notes in Computer Science*, vol. 3198, Sep. 2004, pp. 246-261, doi: 10.1007/978-3-540-30112-7\_21.
- [8] C.A. McGruder, "MVC for Content Management on Cloud," *MS Thesis*, Naval Postgraduate School, 2011.

- [9] Z. Liu, T. Li and G. Yang, "An MVC based Collaborative Framework Support for Test Suite," Proc. of the 14th International Conference on Computer Supported Cooperative Work in Design (CSCWD 10), IEEE Press, Apr. 2010, pp.172-177, doi: 10.1109/CSCWD.2010.5471982.
- [10] J. L. Garrido, M. Gea, N. Padilla, J.J. Canas, and Y. Waern, "AMENITIES: Modelo de entornos cooperativos," Actas del III Congreso Internacional Interacción Persona-Ordenador, pp. 97-104, 2003.
- [11] D. Rodríguez, and R. García-Martínez, "Modeling the Interactions in Virtual Spaces Oriented to Collaborative Work", Chapter 10, Carlos Mario Zapata, Guillermo González, Roberto Manjarrés, Fabio Alberto Vargas, and Wiliam Arévalo (Eds.), Software Engineering: Methods, modeling, and Teaching, Vol. 1, Lima, Perú, 2012.
- [12] A.I. Molina, M.A. Redondo, M. Ortega, and U. Hope, "CIAM. A methodology for the development of groupware user interfaces," Journal of Universal Computer Science, vol. 14-9, 2008, pp. 1435-1446, doi: 10.3217/jucs-014-09.
- [13] V. M. Ruiz Penichet. Task-Oriented and User-Centred Process Model for Developing Interfaces for Human-Computer-Human Environments. Ph.D. dissertation, Universidad de Castilla-La Mancha, 2007.
- [14] M. Van Welie, G.C. van der Veer, and A. Eliëns, "An Ontology for Task World Models," Proc. Of the Eurographics Workshop, Ch.: Design, Specification and Verification of Interactive System, Springer, June 1998, pp. 57-70, doi: 10.1007/978-3-7091-3693-5\_5.
- [15] C. Ellis, and J. Wainer, "A conceptual model of groupware," Proc. of the ACM Conference on CSCW (CSCW 94), ACM, 1994, pp. 79-88, doi: 10.1145/192844.192878.
- [16] M. Anzures-Garcia, L.A. Sanchez-Galvez, M.J. Hornos, and P. Paderewski, "A Knowledge Base for the Development of Collaborative Applications," Engineering Letters, vol. 23-2, Apr. 2015, pp. 65-71.
- [17] M. Anzures-Garcia, L.A. Sanchez-Galvez, M.J. Hornos, and P. Paderewski-Rodriguez, "A Semantic Approach to Develop Groupware," Research in Computing Science: Advances in Computer Science and Engineering, vol. 83, Nov. 2014, pp. 71-83.
- [18] J. Gallardo, C. Bravo, and M.A. Redondo, "A model-driven development method for collaborative modeling tools," Journal of Network and Computer Applications, vol. 35-3, May 2012, pp. 1086-1105, doi: 10.1016/j.jnca.2011.12.009.
- [19] M. Anzures-Garcia, L.A. Sanchez-Galvez, M.J. Hornos, and P. Paderewski, "MVC Design Pattern Based-Development of Groupware", Proc. of the 4th International Conference in Software Engineering Research and Innovation (CONISOFT 2016), IEEE Xplore Digital Library. April 2016, pp. 71-80
- [20] C. Alexander, S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King, and S. Angel, A Pattern Language: Towns, Buildings, Construction. Oxford University Press, New York, 1977.
- [21] C. Alexander, The Timeless Way of Building. Oxford University Press, New York, 1979.



- [22] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal. Pattern-oriented Software Architecture. A System of Patterns, John Wiley and Sons, Vol. 1, 1996.
- [23] S. Huang, and P.-W. Ng, "Essence as a Framework for Conducting Empirical Studies", Chapter 2, Carlos Zapata, Luis Fernando Castro (Eds.), Software Engineering: Methods, modeling, and Teaching, Vol. 3, Medellín, Colombia, 2014
- [24] C.M. Zapata, "An executable pre-conceptual schema for a software engineering general theory", Chapter 1, Carlos Zapata, Luis Fernando Castro (Ed.), Software Engineering: Methods, modeling, and Teaching, Vol. 3, Medellín, Colombia, 2014
- [25] M. Anzures-García, and L.A. Sánchez-Gálvez, "Policy-based group organizational structure management using an ontological approach," Proc. International Conference on Availability, Reliability and Security (ARES 08), IEEE Press, Mar. 2008, pp. 807-812, doi: 10.1109/ARES.2008.186.
- [26] M. Anzures-García, L.A. Sánchez-Gálvez, M.J. Hornos, and P. Paderewski-Rodríguez, "Ontology-Based Modelling of Session Management Policies for Groupware Applications," Lecture Notes in Computer Science, vol. 4739, Feb. 2007, pp. 57-64, doi: 10.1007/978-3-540-75867-98.

# Chapter # 16

## Creating an Estimation Model from Functional Size Approximation Using the EPCU Approximation Approach for COSMIC (ISO 19761)

**Francisco Valdés-Souto**

Science Faculty

National Autonomous University of Mexico (UNAM)

CDMX Mexico City, Mexico

fvaldes@ciencias.unam.mx

### 1. Introduction

Information is acquired gradually throughout the software development life cycle. At the early phase, the majority of information available is at a very high level of abstraction and it is often based on a number of assumptions that can neither be verified nor precisely described at that point in time. This leads to the challenge of having to make decisions on project constraints based on incomplete and, at times, unreliable information [1].

At this early phase of a software development process, management must rely on such incomplete information for decision making purposes, and usually the decisions made were based on estimations. As Tom de Marco defined it: “An estimation is a prediction that is equally likely to be above or below the actual result” [2].

Morgenshtern [3] mentions the usages of project estimation as follows: project selection; staffing; scheduling; monitoring and control; team performance assessment; and marketing.

It is generally recognized that project requirements define the project size (scope), which impact the effort required to develop the project, which then drives the project duration [4].

In the past 40 years, many estimation models and tools have been developed [5]–[24]: the majority of these models focus on estimating effort.

In order to generate estimation models, the researchers have used databases documented on the basis of the past completed projects they participated in, usually, this information is not available to all the persons or is difficult to acquire, or has elements that do not make sense for all the database's users.

Most of the estimation models developed are dependent on the representativeness of the samples (databases) utilized, that is, means, the majority of most of the estimation models must be calibrated locally, aiming at better behavior, and this requires local historical data (a database).

Morgenshtern pointed out that “Algorithmic models need historic data, and many organizations do not have this information. Additionally, collecting such data may be both expensive and time consuming” [3].

In the actual competitive business context, software development organizations are often time-to-market driven. In this context, expert judgment (‘experience-based’) is the estimation approach typically employed in the industry [25].

Of course, there are a number of problems implicit in using the experience to provide estimates, i.e., using measures and estimations still based on researchers’ intuition rather than on rigorous designs and strong experimentation, does not contribute to mature software engineering.

This chapter proposes a solution to generate a historical database in a formal but practical way, utilizing an approximate sizing approach for the COSMIC method [26]. The idea is to reduce the cost and effort required to gather a historical database that could be employed to generate estimation models.

The remainder of this chapter is organized as follows. Section II describes overview information related to estimation models and the importance of databases within this context. Section III presents information of COSMIC method as related to the accepted approximation approaches, their highlights, and mainly the EPCU approximation approach used in this chapter. Section IV describes a quantitative case study with a set of industry projects from a single organization in the financial sector, how a database was created, and how a productivity model can be generated. Section V sets the conclusions.

## 2. Estimation models and databases

It is generally recognized that requirements define the project size (scope), which impact the effort needed to develop it, which then drives the project duration [4] – see Figure 1.

In the past 40 years, many estimation models and tools have been developed [5] – [24]: the majority of these models focus on estimating effort.

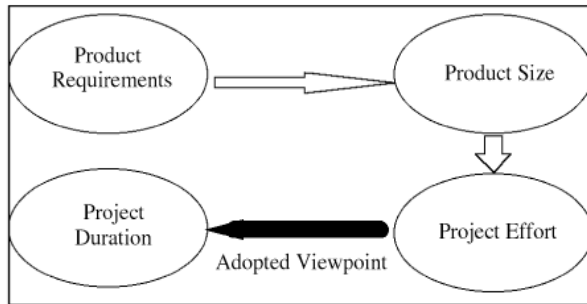


Figure 1. An example of a strategy to estimate project duration [4].

In the literature mainly two distinct approaches of estimation techniques/methods/models classification were found.

The first approach [18, 19, 27], establishes a classification with three categories:

- Expert judgment. This approach was not considered as a technique because the means of deriving an estimate are not explicit. However, the estimation technique typically employed in industry is the one based on the experience of the organization's employees.
- Algorithmic models. These are the most popular ones in the literature. Basically, these models are derived from statistical or numeric analysis over some historical projects.
- Analogy. This was considered a systematic form of the expert judgment approach.

The second approach [18, 19], establish a classification into two major categories:

- Algorithmic models.
- Non-algorithmic models. These were constructed seeking to avoid some of the weaknesses in the algorithmic models and, aimed to model more adequately

the complex relationships between independent variables and the dependent variable.

It is interesting to note that in the previous classifications, except for that of expert judgment, all of the types described utilize mathematical algorithms. This has been considered recently by Abran [28], who defines a common view of the estimation process, identifying only two types of Estimation Models (Figure 2):

- Expert Judgment, and
- Mathematical Model. These models include any models that are derived from statistical, numeric analysis, or more generally, mathematical algorithms.

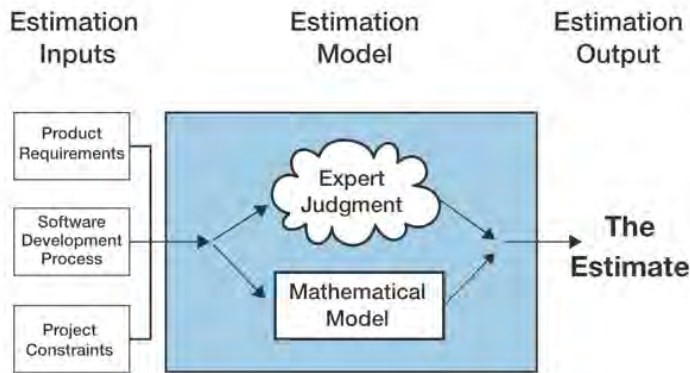


Figure 2. Common view of estimation process [28]

Any estimation model possesses a strong relationship with the measurement process of the input variables employed to generate the estimate. This means that the measurement process comprises the basis of the estimation model: when the measurement of input variables for an estimation model is reliable, greater confidence is generated in the use of the estimation model [1].

In order to generate the estimation models, researchers have used databases documented on the basis of past completed projects they participated in. Usually, this information is not available to everyone, or is difficult to acquire, or it has elements that do not make sense for all the database's users.

The majority of the estimation models developed are dependent on the representativeness of the samples (databases) used.

There is an international database managed by a non-profit organization, the International Software Benchmarking Standards Group (ISBSG). This organization generates reports considering an international sample of data [30].

In the literature there are several estimation techniques, Gray et al. [29], analyze several estimation techniques in terms of modeling capabilities (See Table 1)

In order to acquire a better behavior of the software estimation models developed, most of these need to be calibrated locally, and this requires local historical data (database) documented on the basis of the past completed projects for each organization.

As pointed out by Morgenshtern, “Algorithmic models need historic data, and many organizations do not have this information. Additionally, collecting such data may be both expensive and time consuming” [3].

Of course, for the Mathematical Models defined by [28], this affirmation applies in the same way.

Thus, improvements in software estimation within the context of generating a historical database and estimation models in a less expensive way are welcome in order to contribute to mature software engineering.

## 2.1 Quality Criteria

The more frequent way to compare the estimation models and define the confidence about them, can be called quality criteria: Magnitude of Relative Error (MRE), Standard Deviation (SDMRE) and Prediction level (PRED) – see [14, 17, 18, and 16].

- The Magnitude of Relative Error (MRE) is usually defined by:

$$(1) \quad \text{MRE} = \left| \frac{\text{Actual} - \text{Estimated}}{\text{Actual}} \right|$$

The accuracy of the estimation can also be measured by the Mean Magnitude of Relative Error (MMRE) and the Median Magnitude of Relative Error (MdMRE).

- The SDMRE is usually defined by:

$$(2) \quad MRE = \left| \frac{\text{Actual} - \text{Estimated}}{\text{Actual}} \right|$$

- The prediction level Pred.

$$(3) \quad \text{Pred (I)} = \frac{K}{N}$$

These criteria are very often used in the literature related to estimation models in software. The goal is to gather a low MMRE/MdMRE, a low SDMRE and high PRED.

### 3. Cosmic method

In mature disciplines, it is possible to observe international consensus on measurement, as evidenced by established measurement methods and their respective etalons. In the software domain, there exist international standards only for functional size measurement, including the ISO 14143 series, which prescribe key concepts of the entity and the attributes to be measured. To date, ISO has recognized five (5) Functional Size Measurement (FSM) methods for software that comply with ISO 14143, but only one is referred to as second (2nd) generation of FSM methods: COSMIC – ISO 19761 [31], the current version of this method, is version 4.0.1 and could be found at [www.cosmic-sizing.org](http://www.cosmic-sizing.org).

Table 1. Comparison of estimation techniques in terms of modeling capabilities, adapted from [29].

	Technique	Model Free <sup>1</sup>	Can resist outliers	Explains output	Suits small data sets	Can be adjusted for new data	Reasoning process is visible	Suit complex models	Include known facts
	Least Squares Regression	N	N	P	N	N	Y	N	P
	Robust Regression	N	Y	P	P	N	Y	N	P
	Neural networks	Y	N	N	N	P	N	Y	P
	Fuzzy Systems (Adaptive)	Y	P	Y	Y	P	Y	Y	Y
	Hybrid Neuro-Fuzzy Systems	Y	P	Y	P	P	P	Y	Y
	Rule Based Systems	N	N/A	Y	N/A	N/A	Y	Y	Y
	Case-Based Reasoning	Y	P	Y	P	Y	P	Y	N
	Regression Trees	Y	Y	Y	P	Y	P	Y	P
	Classification or Decision Trees	Y	Y	Y	P	Y	P	Y	P

<sup>1</sup> (Yes = "Y", No = "N", Partially = "P")

### 3.1 COSMIC Approximation

FSM methods work best when the information to be measured – the FURs – is fully known [32].

Santillo [32] further states that the “functional size of software to be developed can be measured precisely [only] after the functional specification stage: this stage is often completed relatively late in the development process.”

However, this is most often not the case in the early phase of software development projects, when detailed information may not be available and when estimations are needed.

Aiming to tackle the problem of obtaining a software size idea when detailed information may not be available (Early Sizing) or when there is not enough time to measure the required software using the standard method (Rapid Sizing), researchers have developed some approximation approaches [31, 33, and 34].

In particular, for the COSMIC method, a Guideline for Early or Rapid COSMIC Functional Size Measurement was released in 2015. In this Guideline the approximation principles and valid approximation approaches for COSMIC were established [26].

Analyzing the approximation approaches described in [26], the following highlights were identified and they are presented in Table 2.

Table 2 shows that the validity of the majority of approximation techniques is dependent on the local calibration of the approximation approaches, except for the EPCU Approximation approach which does not require local calibration and which is useful when there is no historical data available [35, 34].

#### 1. EPCU Approximation approach

The full description and use of the EPCU Approximation approach can be reviewed in [31] and [35]. However, the EPCU model defines an EPCU Context as “a set of variables (inputs and output) and the relations that affect a specific project or a set of similar projects” [1].

For the EPCU Approximation approach as defined in [31] and [35], the EPCU context defined, states the input variables as:



1. Variable 1: the use case/functional process size (evaluated through a subjective, experience-based approach), and
2. Variable 2: the number of objects of interest related to the use case/functional process (also evaluated through a subjective, experience-based approach).

These input variables were utilized in the experimentation in, [31, 34], to approximate functional size for functional process and use cases.

In practice, use cases are wide spread in the industry and can be identified earlier, which is less costly than identifying functional processes [31].

With the EPCU Approximation approach, an organization can collect historical data without the need to perform detailed measurements [35].

**Table 2. Approximation techniques analysis highlights [34].**

Approximation Approach	Needs local calibration	Requirement granularity level	Consideration
Average Functional Process	X	Functional Process	This approximation is valid as long as there is sufficient reason to assume that the sample on which the size of the average functional process is calculated is representative for the software of which the functional size of which size is approximated. [38]
Fixed Size Classification	X	Functional Process	This approximation is valid as long as there is sufficient reason to assume that the assigned size classification is representative for the software of which the functional size of which size is approximated. [38]
Equal Size Bands approximation	X	Functional Process	This method is recommended for the approximate sizing of software where the distribution of the functional process sizes is skewed. For the business application, this method has little added value over the average functional process method (1) or the fixed size classification method (2). [38]
Average Use Case approximation	X	Use Case	This approximation is valid as long as there is sufficient reason to assume that the assigned size classification of an average use case is representative for the software of which the functional size of which size is approximated. [38]

This table continues on the following page →

Approximation Approach	Needs local calibration	Requirement granularity level	Consideration
Early & Quick COSMIC approximation	X	Multilevel Approach (*)	The precision of the method is strongly dependent on the training and capability of the practitioners who use it to understand the categories at higher levels of granularity. [38], this approximation approach combines scaling and classification approaches.
Quick/Early approximation		Use Cases	The precision is directly proportional to the level of granularity of the analyzed use cases model.
EPCU approximation		Functional Process & Use Cases	Does not require local calibration (less expensive) and is useful when there are no historical data available.

## 4. Case study

In order to create an estimation model as described in the proposal, the case study designed in [36] was adapted, with the next steps:

- Gathering projects information
- Defining the database set of projects
- Functional size approximation
- Creating historical database
- Generating effort/cost productivity models
- Creating the Estimation Models

### 4.1 Gathering projects information

As a part of a Spring 2015 teaching course in Project Management for a Master of Science degree, at a Mexican University, a set of eleven (11) projects from a financial institution were gathered, all of the projects are industry projects, meaning they were developed and delivered at the end of 2015. All the projects were in a production environment.

For confidential issues, the projects were labeled as “Project 1”, to “Project 11”, and project selection was conducted considering projects for which the real cost (\$ USD) and effort (person-month) were known (see Table 3).

In Table 3, the first column is the project identifier, the second column is related to the effort in person-months that was utilized to develop the project from requirements to the acceptance test, while the third column is related to the cost required to develop the project, in \$USD.

**Table 3. Data related with effort and cost gathering from all the projects**

Project ID	Effort (person-month)	Cost (\$ USD)
Project 1	3.70	\$ 14,031.41
Project 2	5.20	\$ 18,219.90
Project 3	5.40	\$ 19,371.73
Project 4	5.30	\$ 16,753.93
Project 5	8.40	\$ 30,628.27
Project 6	9.30	\$ 33,246.07
Project 7	9.80	\$ 35,235.60
Project 8	10.50	\$ 34,240.84
Project 9	12.50	\$ 43,560.21
Project 10	16.00	\$ 57,591.62
Project 11	15.00	\$ 52,094.24

## 4.2 Defining the database set of projects

Once the whole projects were identified, two thirds (2/3) were selected randomly to conform the database, this is recommended by Abran in [28].

The projects identified as out of the database were in grey in Table 3 (Project 3, Project 4, Project 7, and Project 11).

With the remaining projects the database was created following the step proposed in [36].

## 4.3 Functional size approximation

With the projects identified as a part of the database, the team leaders who participated in the project's development were asked to perform the following:

1. Identify for each project the use cases related, each use case was labeled (i.e. Use Case 1 to Use Case n).

2. Classify each of the use cases for each project utilizing the linguistic values: Small, Medium, Large and Very Large.
3. Classify the number of objects of interest for each of the use cases for each project utilizing the linguistic values: Few, Average, and Many.
4. Assign values for the two previously classified input variables (points 2 and 3, the use case size, the amount of objects of interest related with the use case) defined from the EPCU context, considering the subjective classification relative functional size of the use cases and the subjective classification on the number of objects of interest in each use case, each value assigned within the range of 0 to 5  $\in$  R.

Considering the information acquired from the team leaders, approximation of the functional size in COSMIC units (CFP) was performed, using the EPCU approximation approach [34].

The COSMIC functional size approximation for each project is depicted in Table 4 and full data considering the approximation size for each use case are presented in Appendix A.

Table 4, illustrates, in the first column, the project identifier, and, in the second column, the COSMIC functional size approximation in CFP using [34].

**Table 4. Cosmic functional size approximation using the epcu approximation approach for each project in database**

Project ID	COSMIC functional size approximation for use case [CFP]
Project 1	76.36
Project 2	127.2
Project 5	308
Project 6	270.04
Project 8	281.9
Project 9	423.88
Project 10	599.52

Considering [34], the quality criteria for the functional size approximation included Mean Magnitude of Relative Error (MMRE) equal to 43% and Standard Deviation of MRE (SD-MRE) equal to 34%. This information is important to know the confidentiality of the EPCU approximation approach that has been studied in previous research and that is out of the scope of this chapter.

## 4.4 Creating historical database

A database that stores data from past projects is fundamental for constructing estimation models [3], or productivity models, as is mention by Abran [28].

The data base is dependent on the representativeness of the samples (databases) used. In addition, the database requires transcendental measurement that enables the use of data over time, techniques, languages, or software development methodologies.

Because in the software domain there exist international standards only for the functional size measurement, the functional size measurement method used is COSMIC – ISO 19761, but employing the EPCU approximation approach, renders data collection efforts and costs to collect data less expensive.

The EPCU approximation approach has been automated by the mechanism “EST-20131211PRXFUC” that could be accessed in [www.mepe.com.mx](http://www.mepe.com.mx). This mechanism was used to make the approximation for each use case for each project in this chapter.

The projects considered derived from the same organization and were developed by the same division. Thus all of the projects possess similar features; for instance, they were developed using Java as the programming language, and they were developed for Web use and Oracle as Database Management System (DBMS). Of course, it could be important to have additional classification characteristics for the projects, in order to enable a more precise selection. For this case study, the projects selected had similar features.

**Table 5. Database created using the epcu approximation approach**

Project ID	COSMIC functional size approximation for use case [CFP]	Effort (person-month)	Cost (\$ USD)
Project 1	76.36	3.70	\$ 14,031.41
Project 2	127.2	5.20	\$ 18,219.90
Project 5	308	8.40	\$ 30,628.27
Project 6	270.04	9.30	\$ 33,246.07
Project 8	281.9	10.50	\$ 34,240.84
Project 9	423.88	12.50	\$ 43,560.21
Project 10	599.52	16.00	\$ 57,591.62

Considering the information acquired, the basic attributes for the database are described as follows:

- Project ID
- COSMIC functional size approximation for use case
- Effort (person-month)
- Cost (\$ USD)

The database is presented in Table 5. This comprises a merger of Tables 3 and 4 (see the column description).

More attributes for the database must be defined if the projects have distinct features (technical and environmental). It is relevant that the database attributes defined enable a classification of projects in a consistent way.

## 4.5 Generating effort/cost productivity models

Once the database was created, a productivity model could be generated. Abran mentions that "...the productivity model represents the modeling of the relationship between the two variables ..., that is between the independent variables (the size of the software) and the dependent variable (completed project effort)" [28].

The productivity models were mathematical models constructed using data from completed projects, and the productivity models will be utilized as estimation models [28].

In order to define a productivity model, a scatter point that correlates the functional size approximation (x axis) with cost (y axis) is generated (figure 3).

From the scatter point and using Excel it is possible to obtain the productivity model with a correlation function represented by:

$$(4) \quad y = 82.827 x + 8381$$

With a correlation factor of  $R^2 = 0.9813$ . In this case, there is a high  $R^2$ , representing a good correlation between independent variable (functional size in COSMIC units) and dependent variable (cost).

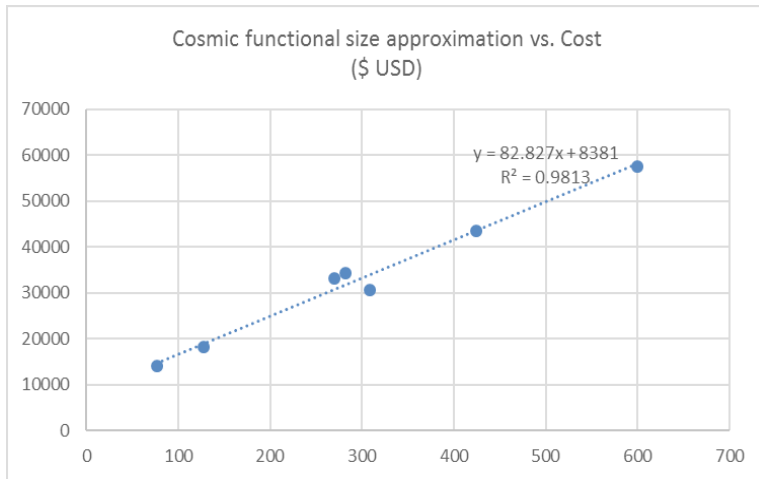


Figure 3. Scatter point functional size approximation vs. cost

Considering the scatter point correlating functional size approximation (x axis) with effort in person-month (y axis) shown in figure 4, a productivity model could be obtained using Excel.

The productivity model is defined by the correlation function:

$$(5) \quad y = 0.0233x + 2.4288$$

With a correlation factor of  $R^2 = 0.957$ . In this case, there is also a high  $R^2$ , representing a good correlation between independent variable (functional size in COSMIC units) and dependent variable (effort).

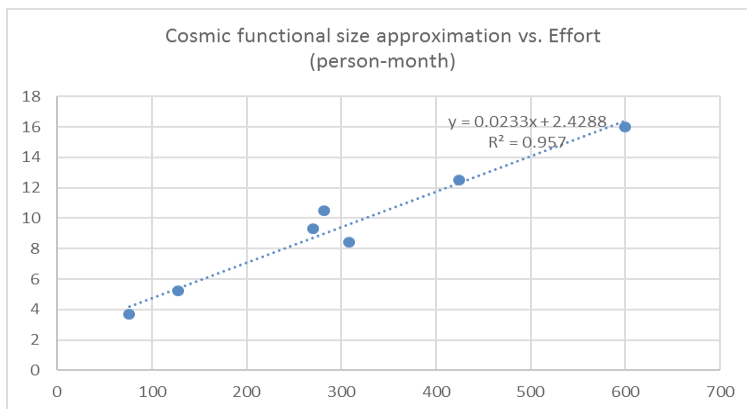


Figure 4. Scatter point functional size approximation vs. effort

## 4.6 Creating the Estimation Models

Once the productivity models (effort/cost) were generated, it is possible to represent the relationship between the independent variables (the size of the software) and the dependent variable (completed project effort) for past projects [28].

In order to determine the estimation models for effort and cost, it is needed to evaluate the confidence of the productivity models related to a distinct set of projects that were not included in the database.

For this issue, the projects identified as out of the database were used to be evaluated in order to obtain a cost estimation and effort estimation with (4) and (5) respectively.

The results obtained in the estimation were shown in Table 6.

**Table 6. Estimation from the productivity models using projects out of the database**

Project ID	COSMIC functional size approximation for use case [CFP]	Effort Estimation (man-month)	Cost Estimation (\$ USD)	Effort MRE (respect the real effort, Table 3)	Cost MRE (respect the real cost, Table 3)
Project 3	161.24	6.19	21,736.03	14.5%	12.2%
Project 4	153.60	6.01	21,103.23	13.4%	26.0%
Project 7	231.73	7.83	27,574.50	20.1%	21.7%
Project 11	522.44	14.60	51,653.14	2.7%	0.8%

In Table 7, it is possible to observe that effort estimation and cost estimation were closer to the real value.

Related to the effort estimation the highest MRE is 20.1% (for Project 7) and the lowest MRE is observed in Project 11 (2.7%).

Considering the cost estimation, the lowest MRE is for the Project 11 (0.8%), and the highest MRE is observed for the Project 4 (26%).

### 1. Calculating the estimation model confidence

The way to compare and to know the confidence for an estimation model, is by calculating the quality criteria, the Table 8, depicts the quality criteria for both estimation models.



Table 8. Quality criteria

	Effort MRE	Cost MRE
MMRE	12.7%	15.2%
SDMRE	7.3%	11.2%
Pred(25%)	100.0%	75.0%

For the estimation models created, the MMRE is less than 16% (12.7% and 15.2% for Effort and Cost respectively), also the SDMRE for both estimation models is low (7.3% and 11.2%).

On the other hand, the Prediction level (PRED (25%)) is high for both estimation models. While the quality criteria for this case study were good, this could have happened when the projects were similar in features, i.e. technology, business area, etc.

In the case study developed in [36], the objective was to show how a database could be created in a practical way with a low cost and effort, and further work was defined as “the assessment of the productivity models generated”. In this chapter, the assessment of the productivity models as estimation models was made using the more often used quality criteria in the literature.

## 5. Conclusions

In previous papers [31, 34, 35] the usefulness of the EPCU approximation approach was reviewed. Some of the advantages of this approximation approach were the following:

- Does not require local calibration and is useful when there are no historical data available.
- Is less expensive to calibrate than the other approximation approaches, which require historical data.
- Exhibits good behavior, even when the individual is not acquainted with the COSMIC method.
- Exhibits good behavior, even when the requirements were not fully known.
- Enables systematic replication of the information.

In this chapter, eleven (11) industry projects, which were not measured using any FSM method, were considered for gathering their COSMIC functional size via the approximation approach.

With the functional size approximated for seven projects, productivity models that correlate functional size with the resource data of the projects (effort and cost), were generated. The productivity models were used to generate estimation models. [28]

With this practical way to determine the functional size for past projects, the need to have a database and formal estimation model could be tackled. This reduced the time for acquiring a confidence database and estimation models for organizations.

Once the database was created and after it had been used by organizations, improvements such as employing the full COSMIC method, or adding new projects could be integrated incrementally, and it will improve the estimation results.

In this chapter, the benefits of the EPCU approximation approach were exploited in order to create a database and estimation models with low cost and effort, because there is no need for a local calibration and the utilization of the EPCU approximation approach is automated in a mechanism that could be accessed via [www.mepe.com.mx](http://www.mepe.com.mx), reducing the calculation effort and the COSMIC knowledge required.

Because the EPCU model is a formal model, that has been tested for approximation of COSMIC functional size [31, 34, 35], and because it enabled the creation of database and estimation models with a low cost and effort, the use of the approach proposed in this chapter contributes to mature software engineering.

## 6. Further work

This chapter proposes a solution for generating historical database and estimation models in a formal but practical way, utilizing an approximate sizing approach by means of the COSMIC method, keeping in mind the reduction of the cost and effort required to gather a historical database.

An interesting work could be utilizing a functional measurement with the standard COSMIC method in order to compare the real functional sizes against the approximated ones.

## 7. References

- [1] F. Valdés-Souto, Design of A Fuzzy Logic Estimation Process for Software Projects: Estimation of Projects in a Context of Uncertainty EPCU Model. Germany: LAP LAMBERT Academic Publishing, 2012.

- [2] T. de Marco, *Controlling Software Projects*. Englewood Cliffs, N.J.: Prentice Hall, 1982.
- [3] O. Morgenshtern, T. Raz, and D. Dvir, "Factors Affecting Duration and Effort Estimation Errors in Software Development Projects," *Inf. Softw. Technol.*, vol. 49, no. 8, pp. 827–837, 2007.
- [4] B. F. Bourque, Pierre, S. Oligny, A. Abran, "Developing Project Duration Models in Software Engineering," *J. Comput. Sci. Technol.*, vol. 22, pp. 348–357, 2007.
- [5] P. K. Suri and P. Ranjan, "Comparative Analysis of Software Effort Estimation Techniques," *Int. J. Comput. Appl.*, vol. 48, no. 21, pp. 12–19, 2012.
- [6] K. Pillai and V. S. Nair, "A Model for Software Development Effort and Cost Estimation," *Softw. Eng. IEEE Trans. Softw. Eng.*, vol. 23, no. 8, pp. 485–497, 1997.
- [7] L. H. Putnam, "A General Empirical Solution to the Macro Software Sizing and Estimating Problem," *IEEE Trans. Softw. Eng.*, vol. SE-4, no. 4, pp. 345–361, 1978.
- [8] A. J. Albrecht and J. E. Gaffney, "Software Functions, Source Lines of Codes and Development Effort Prediction: A Software Science Validation," *IEEE Trans. Softw. Eng.*, vol. 9, no. 11, pp. 639–648, 1983.
- [9] T. K. Abdel-Hamid and S. E. Madnick, "Impact of Schedule Estimation on Software Project Behavior," *IEEE Softw.*, vol. 3, no. 4, pp. 70–75, 1986.
- [10] T. K. Abdel-Hamid, "Dynamics of Software Project Staffing: A system Dynamics Based Simulation Approach," *IEEE Trans. Softw. Eng.*, vol. 15, no. 2, pp. 109–119, 1989.
- [11] T. Mukhopadhyay, S. S. Vicinanza, and M. J. Prietula, "Examining the Feasibility of a Case-Based Reasoning Model for Software Effort Estimation," *MIS Q.*, vol. 16, no. 2, p. 155, 1992.
- [12] R. Madachy, "A Software Project Dynamics Model for Process Cost , Schedule and Risk Assessment," University of Southern California, 1994.
- [13] K. Srinivasan and D. Fisher, "Machine Learning Approaches to Estimating Software Development Effort," *IEEE Trans. Softw. Eng.*, vol. 21, no. 2, pp. 126–137, 1995.
- [14] M. Shepperd, C. Schofield, and B. Kitchenham, "Effort Estimation Using Analogy," in *Proceedings of IEEE 18th International Conference on Software Engineering*, 1996, pp. 170–178.
- [15] G. K. Michelle, M. Cartwright, L. Chen, and M. J. Shepperd, "Experiences Using Case-Based Reasoning to Predict Software Project Effort," in *Proceedings of the 4th International conference on empirical assessment and evaluation in software engineering*, 2000, no. M1, pp. 1–22.
- [16] I. Myrtveit and E. Stensrud, "A Controlled Experiment to Assess the Benefits of Estimating with Analogy and Regression Models," *IEEE Trans. Softw. Eng.*, vol. 25, no. 4, pp. 510–525, 1999.
- [17] A. Idri and A. Abran, "Towards A Fuzzy Logic Based Measures for Software Projects Similarity," in *MCSEAI'2000 - Maghrebian Conference on Computer Sciences*, 2000, pp. 1–12.

- [18] A. Idri and A. Abran, "Fuzzy Analogy: A New Approach for Software Cost Estimation," in 11th International Workshop on Software Measurement (IWSM'01), 2001, pp. 93–101.
- [19] A. Idri, A. Abran, and T. M. Khoshgoftaar, "Estimating Software Project Effort by Analogy Based on Linguistic Values," in Proceedings - International Software Metrics Symposium, 2002, vol. 2002-Janua, pp. 21–30.
- [20] R. Jeffery, M. Ruhe, and I. Wieczorek, "Using Public Domain Metrics to Estimate Software Development Effort," in Seventh International Software Metrics Symposium. METRICS 2001, 2001, pp. 16–27.
- [21] P. Efe and O. Demirors, "Applying EVM in a Software Company: Benefits and Difficulties," in 2013 39th Euromicro Conference on Software Engineering and Advanced Applications, 2013, pp. 333–340.
- [22] Barry.W. Boehm, *Software Engineering Economics*, 1st ed. Englewood Cliffs, NJ, USA: Prentice Hall, 1981.
- [23] B. S. Barry W. Boehm, Chris Abts, A. Winsor Brown, Sunita Chulani, Bradford K. Clark, Ellis Horowitz, Ray Madachy, Donald J. Reifer, *Software Cost Estimation with COCOMO II*, 1st ed. Upper Saddle River, NJ, USA: Prentice Hall, 2000.
- [24] F. Valdés-Souto and A. Abran, "Industry Case Studies of Estimation Models Using Fuzzy Sets," in *Software Process and Product Measurement, International Conference, IWSM-Mensura 2007*, 2007, pp. 87 – 101.
- [25] J. Gómez, "El Laboratorio de las TI," *Resultados del Estudio sobre Medición del Tamaño del Software 2014*, 2015. [Online]. Available: <http://www.laboratorioti.com/2015/02/11/resultados-del-estudio-sobre-medicion-del-tamano-del-software-2014/>. [Accessed: 10-Feb-2016].
- [26] Common Software Measurement International Consortium (COSMIC)., "Guideline for Early or Rapid COSMIC Functional Size Measurement," 2015.
- [27] M. Shepperd and C. Schofield, "Estimating Software Project Effort Using Analogies," *Softw. Eng. IEEE Trans. Softw. Eng.*, vol. 23, no. 12, pp. 736–743, 1997.
- [28] A. Abran, *Software Project Estimation: The Fundamentals for Providing High Quality Information to Decision Makers*, 1st ed. Hoboken, NJ, USA: John Wiley & Sons, 2015.
- [29] A. R. Gray and S. G. MacDonell, "A Comparison of Techniques for Developing Predictive Models of Software Metrics," *Inf. Softw. Technol.*, vol. 39, no. 6, pp. 425–437, 1997.
- [30] C. Symons, "The Performance of Real-Time, Business Application and Component Software Projects," 2011.
- [31] F. Valdés-Souto and A. Abran, "COSMIC Approximate Sizing Using a Fuzzy Logic Approach: A Quantitative Case Study with Industry Data," in *2014 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement*, 2014, pp. 282–292.

- [32] L. Santillo, "Early and Quick COSMIC FFP Overview," in *COSMIC Function Points: Theory and Advanced Practices*, A. A. Reiner Dumke, Ed. Boca Raton, FL, USA: CRC Press, 2011, pp. 176–191.
- [33] R. Meli, "Early Function Points: A New Estimation Method for Software Projects," in *ESCOM 1997*, 1997, pp. 1–10.
- [34] F. Valdés-Souto and A. Abran, "Improving the COSMIC Approximate Sizing Using the Fuzzy Logic EPCU Model," in *Joint Conference of the 25rd International Workshop on Software Measurement & 10th International Conference on Software Process and Product Measurement - IWSM-MENSURA 2015*, 2015, pp. 192–208.
- [35] F. Valdés-Souto and A. Abran, "Case Study: COSMIC Approximate Sizing Approach Without Using Historical Data," in *Joint Conference of the 22nd International Workshop on Software Measurement and the 2012 Seventh International Conference on Software Process and Product Measurement*, 2012, pp. 178–189.
- [36] F. Valdés-Souto, "Creating a Historical Database for Estimation Using the EPCU Approximation Approach for COSMIC (ISO 19761)," 4th Ed. *Int. Conf. Softw. Eng. Res. Innov.*, no. Iso 19761, 2016.

# Chapter # 17

## A Representation Based in SEMAT Kernel of the Test Planning Process According to ISO/IEC/IEEE 29119-2 Standard

**Fabio Alberto Vargas Agudelo,  
Dario Enrique Soto Duran,  
Juan Camilo Giraldo Mejia**

Grupo de Investigación en Ingeniería de  
Software del Tecnológico de Antioquia  
- GIISTA  
Tecnológico de Antioquia  
Medellín, Colombia  
{fvargas,dsoto,jgiraldo1}@tdea.edu.co

**Claudia Elena Durango Vanegas**

Grupo de Modelamiento y Simulación  
Computacional GIMSC  
Universidad de San Buenaventura  
Medellín  
Medellín, Colombia  
claudia.durango@usbmed.edu.co

### 1. Introduction

Software engineering seeks standardization and normalization processes and a common ground of key elements that constitute a kernel, providing to software analysts use it in different phases of the lifecycle, as a set of methods and best practices to ensure the software product quality. Quality assurance is the purpose of software testing, performing verification and validation during the lifecycle of its construction [1].

In [2], it states that quality is the degree of a set of features meets the requirements, indicating satisfaction in the software product by stakeholder. SEMAT includes a kernel widely accepted by the software development community, extensible to other specific uses simple language to describe methods and best practices, looking for standardization and normalization process elements. Test Planning Process applies mainly to develop the Test Plan. Depending on the specific phase or extent of the test to evaluate, it can be a Project Test Plan or a Test Plan. This chapter proposes a representation the SEMAT kernel of thread Test Planning Process of Organizational Process of ISO/IEC/IEEE 29119-2 Standard [3].

This chapter is organized as follows: Section 2 presents the most relevant concepts of SEMAT, management of software testing and Planning Software Test; Section 3 descri-

bes Planning Software Test thread SEMAT by a representation based on SEMAT kernel; the conclusion and future work relate to this experience are provided in Section 4.

## 2. Theoretical Framework

### 2.1 SEMAT (Software Engineering Method and Theory)

SEMAT is based on a strong theory, with approved principles and best practices to support the process of redefining software engineering. It includes a kernel of elements and simple language to describe methods and best practices. The kernel contains a small number of “things we always work with” called alphas: opportunity, stakeholders, requirements, software system, work, team and way to working, and “things we always do” called activity spaces like to explore possibilities, understand stakeholder needs, prepare to do the work, and others, for developing system software (Figure 1) [4]. SEMAT allows to organize interesting aspects in three areas customer (green), solution (yellow) and endeavor (blue). Additionally, it involves alphas, activity spaces, activities, best practices, methods, patterns and work products (see Table 1) [5].

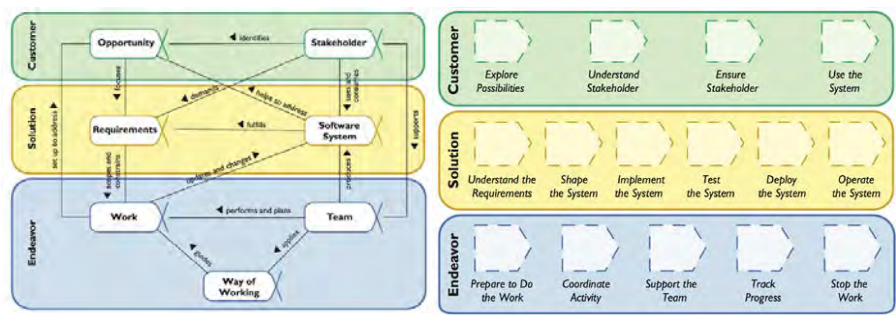




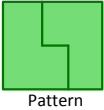



Figure 1. Alphas and Activity Spaces in software system development process [6]

Table 1. Semat Core Elements [5]

Element	Description	Element	Description
	Describes things that a team should manage, produce and use in the process of development, maintenance and support. SEMAT identifies seven alphas (Figure 1).		Represents the essential things should be performed to develop the software. SEMAT defines 15 activity items (Figure 1).

This table continues on the following page →

	<p>Defines one or more types of work products and one or more work types, as well as giving guidance on how to use them in a practical context.</p>		<p>This is a group of SEMAT kernel elements necessary to express the work guide with a specific target.</p>
	<p>This is a description of a practical structure.</p>		<p>This is an artifact of value and relevance to the effort of software engineering. A work product can be a document, a piece of software, a software test or the preparation of a training.</p>

## 2.2 Software Test

In literature, there are various definitions of Software Testing like “activity in which a system or component runs under controlled conditions, the results are recorded and the evaluation is done on some aspect of the system or component” [7]. ISO/IEC/IEEE 29119 standard software testing is defined as “test planning, controlling, analyzing, designing, implementing and executing. It also allows the evaluation of reporting representation, and closing software test” [8].

Software testing is focused exclusively on the lifecycle in order to ensure that the product is built according to the design [8]. Within the steps of software testing are preparation, planning, execution, analysis, and monitoring [9]. This corresponds to tests as well as development software product has a lifecycle that characterizes them and immediately initiated the development of the project plan starts with the specification requirements [10]. Implementing the lifecycle testing from test plan based on the project plan. Following this and based on the requirements specification the tests are performed [11]. Consequently, software testing is considered as a parallel to the development process that allows evaluation process from different aspects and behavior of a system or component.

## 2.3 ISO/IEC/IEEE 29119-2 Standard

Testing process is supported on the ISO/IEC/IEEE 29119-2 Standard covers the processes of organization, management level testing, and dynamic level of evidence. This standard has a testing approach based on risks, corresponding to a recommended method for defining the strategy to focused and prioritized tests [2].



## 2.4 Organizational Test Processing

The process of testing organization, a testing policy is established in order to express the expectations of organization management and focus on commercial terms of software, as well as allowing involve people outside the test as executives and managers. The test policy is a guide in formulating strategic and organizational performance in the testing process. The strategy identifies the requirements and limitations of processes at management level and dynamics of the evidence. The Organizational Test Process is generic, and directs the strategy project test to run (Figure 2) [3]. Examples of Organizational Test are policies, strategies, processes, procedures, and others test assets of the organization.

## 2.5 Test Planning Process

Planning process allows establish scope, resources, strategy and, activities testing from a risk-based approach. The purpose of the process is to define the test plan of the project or the specific phase of the product. The plan is an input essential to ensure the success of the project, it allows to establish a base line for implementation, monitoring and control.

Planning includes the activity of understanding the context. In this instance, relevant information is searched in order to apply the tests. The purpose of the test is to identify requirements, models and software devices to be evaluated taking into account the verification and validation processes that integrate testing concept. To extend the scope of understanding the context should know the assets of organizational processes and environmental factors that affect the testing project. Organizational process assets refer to documents such as organizational policies and strategies testing. Environmental factors relate to standards, software development methodologies and regulations applicable to the business application framework.

Once the context is understood, the development of test plan begins by identifying the factors for the success of the tests, defining a test plan that integrates the following elements: scope, risks, types and test techniques, actors, test metrics, effort and time estimates, tools, and schedule.

The goal of the tests is ensuring the stability of the business instead of test the possible scenarios of the software. The essential factor in the success of the test is to identify risks that compromise rules and business processes, it means the risks assumed from business with possible defects that may have the software in production. Therefore, it must make an identification, characterization, analysis, and risk mitigation.

To identify the risks of a software project, the sources that originate should take into account such as [12].

- Engineering Product: These risks are oriented to the technical aspects of the work to be done. This type of risk refers to the risks facing the business at the time that the programs have an error in production.
- Development environment: refers to the methods, procedures and tools used to make the product.
- Program restrictions: contractual, organizational and operational factors that affect the software developed.

The design of a successful test strategy from of an appropriate risk analysis taking into account the following variables: probability and impact. This analysis generates the risk prioritization and allows for a mitigation approach. Mitigation should integrate elements like factors or relevant software criteria and products to be tested.

Testing Strategy should determine the scope of the test, test type, characteristics to be tested, test techniques, completion criteria, suspension and resumption of testing. The strategy should consider restrictions in the project, product and organization, to conform the bases to project management, scope, time, and cost. Consequently, the plan of human resources required for execution of the strategy and schedule of activities with their respective principals are established. Finally, the test plan must be approved by the team that participated in its design and disseminated different project stakeholders. The above process can be verified graphically in Figure 3.

### 3. Background

According to [13], associated elements are identified with the process of validation and verification of software quality ISO / IEC / IEEE 29119-2 Standard. The main elements are associated with best practices of RUP “Verify software quality”, the alpha “Software System”, Work Products of Test Planning Process (Figure 4). This chapter does not describe a representation of the Test Planning Process.

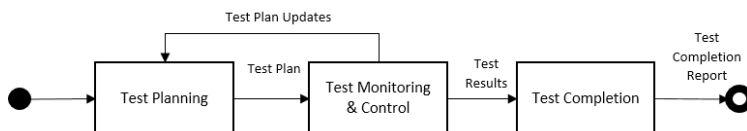


Figure 2. Activity of Organizational Test Process [3]

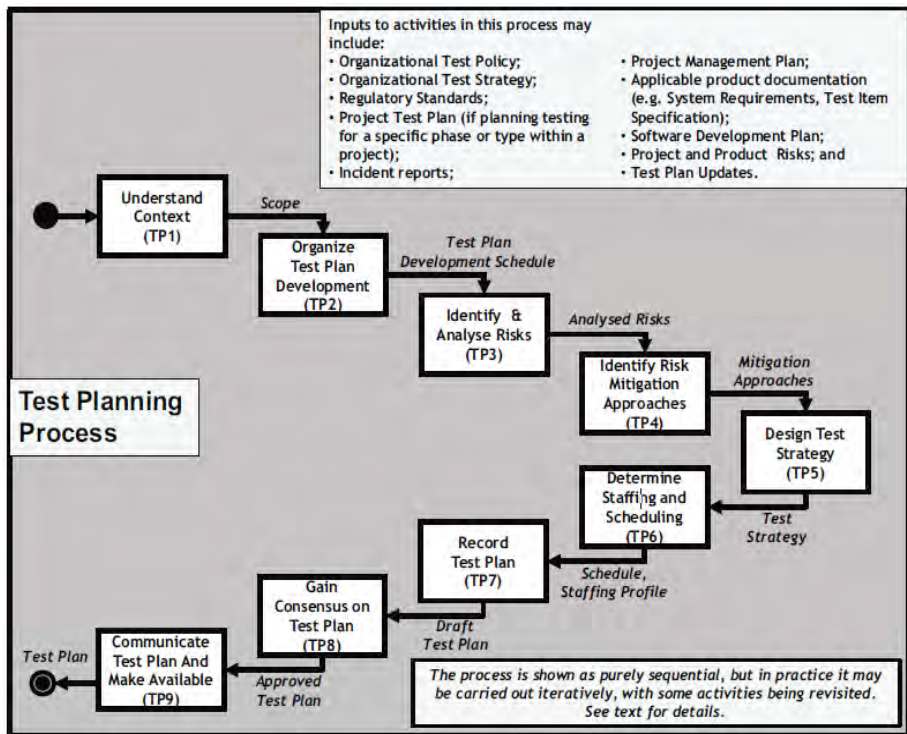


Figure 3. Test Planning Process [3]

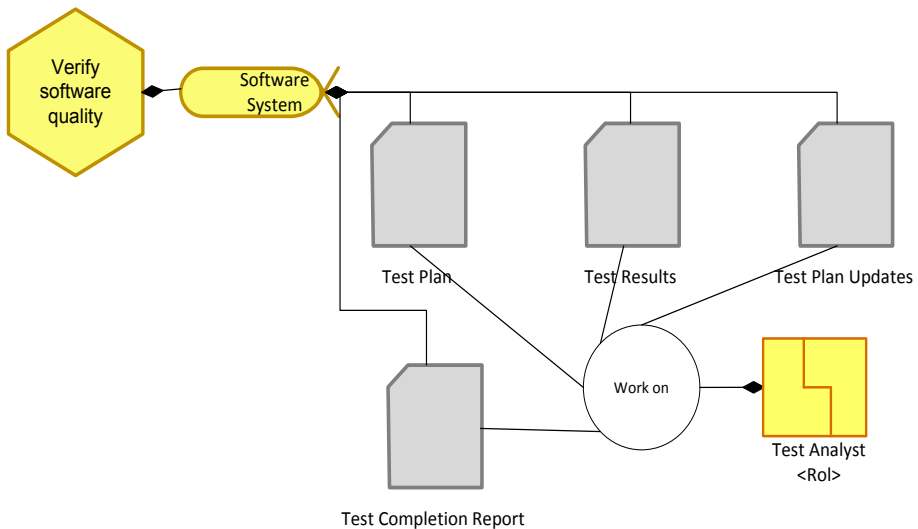


Figure 4. Phase Test Process Model [13]

## 4. Representation based in SEMAT Kernel of Test Process Model of ISO/IEC/IEEE 29119-2:2013 Standard

A relevant aspect of representation is identifying the constant elements of the Test Process Model in SEMAT Kernel. These elements are best practices, area of interest, alphas, activity and, activity spaces.

### 4.1 Best Practice

RUP identifies best practices and incorporates six best practices of software in processing model: requirements management, component-based architecture, visual model software, check software quality, change control software and software development architecture iteratively (Figure 5). In our project, we focus on “Verify Software Quality”.

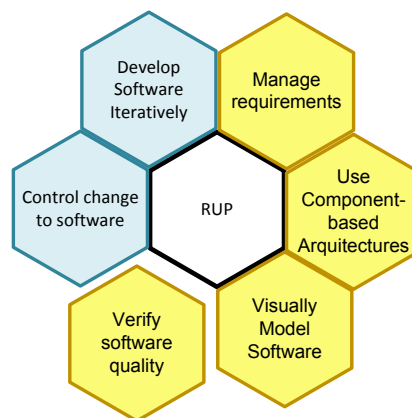


Figure 5. SEMAT kernel representation in six RUP best practices [13]

### 4.2 Concern Areas

Test Process Model focuses on the area of interest “Solution”. This area contains the elements that relate to the specification and development of software system. The elements found in this area are of color yellow (Figure 1).

### 4.3 Alpha

Test Process Model is associated with alpha “Software System” because these tests are associated with software to develop (Figure 6) [13].

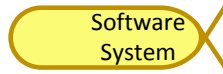


Figure 6. SEMAT Alfa representation [13]

## 4.4 Areas of activity

Next step is to identify the activities and work products of Test Planning Process. Below this activity is shown in Table 2.

Table 2. Activity and work product of Test Planning Process of ISO/IEC/IEEE 29119-2:2013 Standard

Activity	Work Product
Understand context	Test Requirements Communicate Plan
Organize Test Plan Development	Scope statement (WBS) Registration Actors
Identify and organize Risks	Risk Plan
Design Test Strategy Determine Staffing and Scheduling	Specification of scope (Schedule, staffing profile)
Record Test Plan Gain Consensus on Test Plan Communicate Test Plan and Make Available	Test Plan

In order to make the representation Table 2 with phases (Figure 7), best practices and identified alpha [13] are used. In the representation are considered the elements identified in Table 2. The practice “Verify software quality” is associated with the phases of the ISO/IEC/IEEE Standard 29119-2: 2013 [13], where the space activity “Test the System” and activities phases are found.

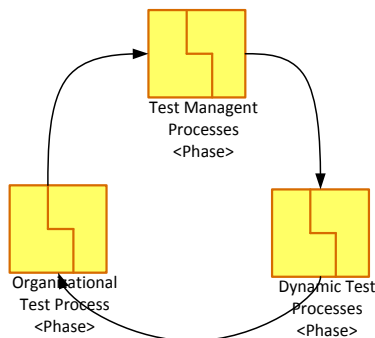


Figure 7. SEMAT core representation ISO/IEC/IEEE 29119-2:2013 Standard [13]

In Figure 8 shows Process Test phase containing space activity to test the system and activities. In Table 2 is shown these elements are associated with the practice “Verify Software Quality”.

In Figure 9 shows, the role Test Analyst works on all products in work seen in Table 2, there are also related to alpha “Software System” of the best practice “Verify Software Quality”.

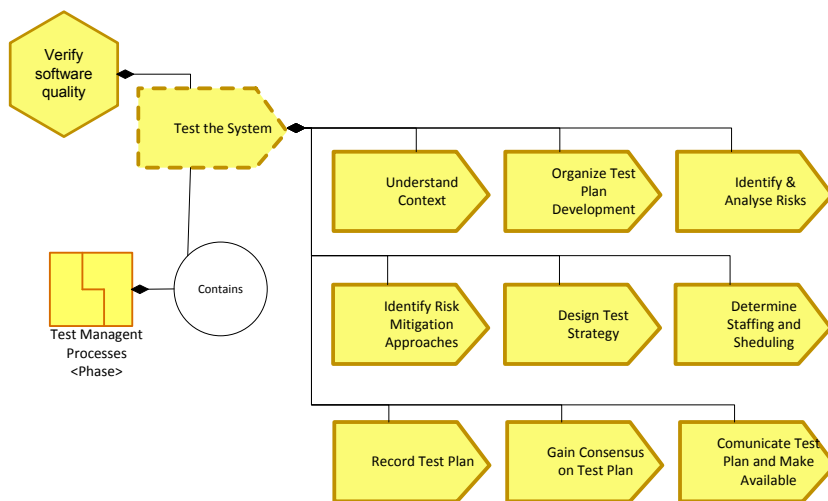


Figure 8. SEMAT core representation of activities of Test Planning Process

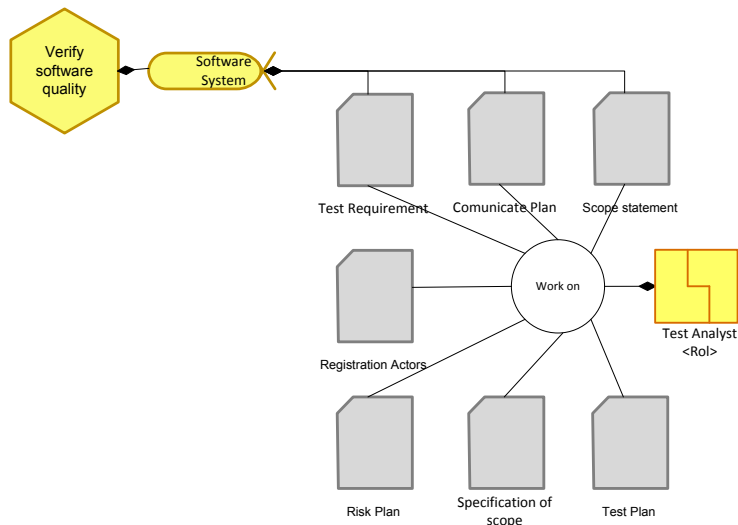


Figure 9. SEMAT core representation of work products from Test Planning Process

## 5. Conclusions and future work

To make the process of software testing is relevant to identify and mitigate risks in order to define suitable strategies on context and ensure the stability of the product in the production environment.

The representation of the planning process of testing on SEMAT kernel describes the activities and work products that serve for supporting to execute and monitoring the test project.

SEMAT Kernel representation of the ISO/IEC/IEEE Standard 29119-2: 2013 allows the users a better understanding of the elements that comprise it. So, this representation allows identify and follow a sequential order of phases for implementation, and the activities to develop products associated in each activity. Another aspect to highlight is Test Analyst to obtain the expected results.

The representation allows demonstrating traceability and consistency in practice verification software quality with the phases of the rules, where the products work and the role associated.

## 6. References

- [1] Myers, G. J., Sandler, C., Badgett, T., & Thomas, T. M. "The Art of Software Testing". New Jersey, USA: Wiley, 2004.
- [2] ISO. Norma internacional ISO 9000 - Sistemas de gestión de la calidad - Fundamentos y vocabulario. Ginebra: ISO 9000, 2005.
- [3] IEEE, International Standard ISO/IEC/IEEE 29119-2: Software and systems engineering Part 2: Test Processes. 2013, pp. 1–68.
- [4] I. Jacobson, P. Ng, and P. E. McMahon, "La Esencia de la Ingeniería de Software: El Núcleo de SEMAT". vol. 1, no. 3, pp. 71–78, 2013.
- [5] C. E. Durango Vanegas and C. M. Zapata Jaramillo, "Una representación basada en SEMAT y RUP para el Método de Desarrollo SIG del Instituto Geográfico Agustín Codazzi" Ing. USBmed, vol. 6, no. 1, pp. 24–37, 2015.
- [6] M. J. Simonette, L. Lago, and E. Spina, "Extending SEMAT kernel to deal with developer error," Int. J. Circuits, Syst. Signal Process., vol. 8, pp. 253–258, 2014.
- [7] International Standard ISO/IEC/ IEEE 24765 First edition, 2010.
- [8] Glenford, J. Wiley, J. The art of software testing, 2004.
- [9] Tian, J. Software Quality Engineering: Testing, Quality Assurance, and Quantifiable Improvement. Chichester, England: Wiley-IEEE Press, 2005.

- [10] Sommerville, I. Ingeniería del software. Madrid: Pearson. Séptima edición, 2005.
- [11] Barranco, J. Metodología del análisis estructurado de sistemas. España: Comillas, 2001.
- [12] Medina, Y. 2009. Taxonomía Ajustada para la Identificación de Riesgos en los Proyectos de Desarrollo de Software de la Universidad de las Ciencias Informáticas. [Tesis] Ciudad de la Habana: UCI, 2009.
- [13] Vargas, F.A. Soto, D.E, Durango, C.E, Giraldo, J.C. Representación en el Núcleo de SEMAT de la Norma ISO/IEC/IEEE 29119-2 para identificar patrones en Pruebas de Software. 4to. Congreso Internacional de Investigación e Innovación en Ingeniería de Software, CONISOFT, 2016.



# Chapter # 18

## A KAOS representation by using the SEMAT kernel

**Luis Fernando Castro Rojas, Santiago Montaña Lince, and Esperanza Espitia Peña**

Departamento de Ingeniería de Sistemas y Computación  
Universidad del Quindío  
Armenia, Colombia  
{lufer, slincem, eespitia}@uniquindio.edu.co

### 1. Introduction

The requirements analysis includes the specification and validation of the services to be provided by the system, as well as the operational constraints [8,15]. In order to complete this analysis, several methods have been implemented. Some of these methods are focused on the goal-oriented requirement specification: TROPOS, I\* [18], KAOS (Knowledge Acquisition Automated Specification) [2,4,10], and GBRAM, which are suitable to be represented by using the SEMAT (Software Engineering Method and Theory) kernel [13]. These methods define software requirements from organizational objectives, and provide a framework for relating organizational goals and problems in the project formulation for making decision [12].

The aforementioned approaches exhibit some shortcomings related to multiple interpretations which are caused by ambiguity, overgenerality, synonymy, and vagueness [1]. An analysis of KAOS revealed several limitations associated with redundancy, overload, incompleteness, and lack of definitions [3]. Babar, Wong, and Gill in [6] present a study based on five major goal-oriented approaches (i\*, KAOS, GBRAM, NFR, MAP). The authors exhibit two fundamental weaknesses presented by such approaches, which limit their practical use: (1) strategic alignment concepts are misunderstood in the context of the information system management discipline and (2) a critical analysis of the goal-oriented approaches in order to assess their suitability for modeling strategic alignment concepts is absent.

Additionally, these methods lack an adequate representation and specification of the organizational objectives, problem domain and system objectives [5].

Nowadays, the software industry revolves around the standardization of processes and the union of elements under a common core [13]. SEMAT is a structured framework that includes a kernel for establishing a common ground for the methods and practices contained therein. Therefore, we propose the representation of KAOS by using the SEMAT kernel, in order to provide a theoretical foundation related to GORE methods.

The Chapter is organized as follows: In Section II we describe the theoretical aspects of the GORE methods, especially KAOS. In Section III we describe a proposal for the inclusion of KAOS in the SEMAT kernel. Finally, in Section IV we present some conclusions and future work.

## **2. Theoretical aspects of the gore methods**

### **2.1 GORE METHODS**

Gore (Goal Oriented Requirement Engineering) methods are a set of approaches that promotes the use of goals as the basis for the software requirements, including, in this way, a point of view intentional, that is, the purpose of the system. The introduction of a point intentional view allows stakeholders to express their needs in a more natural manner, focusing on what they want (your goals) versus how to achieve them (conventional requirements). From the goals, requirements can be derived as ways to achieve these goals [13].

### **2.2 TROPOS**

This proposal, based on *I star* represents a methodology for the organization modeling, widely used in the early processes of software requirements elicitation. This methodology allows to capture not only the what? Or the how? , but also the why? Of software development in the organization. This methodology realizes a detailed description of the system dependencies develop and achieve adequate specification of functional and nonfunctional requirements [11,13].

### **2.3 I\*- I star**

It is a modeling framework of organizational contexts based on the dependency relationships between the actors. The main idea is that the actors depend on each other to achieve the goals, and to provide the resources required for doing the tasks and for accomplishing the goals [9]. In this framework, an actor is an active entity that performs

actions to achieve goals through the exercise of “know-how” and an actor is considered a “super class” to agent, position and role [9, 13].

This goals oriented language includes nodes that represent actors, goals, tasks and resources, addition to a set of relationships between them. Also, it is an approach that uses the idea of softgoal. The main feature of the business modeling on other fields of requirements engineering is the importance of the agents. An agent is defined as an entity that exists in an organization, that has goals and that can perform tasks or use resources to achieve those goals or help other agents achieve their goals [13].

## 2.4 GBRAM (Goal-Based Requirements Analysis Method)

GBRAM is a goal-based approach for identifying, elaborating, refining and organizing goals for requirements specification. GBRAM considers the initial identification and abstraction of goals for all available information sources. Lastly, the goals are translated into operational requirements by generating a specification requirement document (SRD) [13].

The GBRAM method includes two types of activities: goal analysis and goal refinement. Goal analysis encompasses the exploration of documentation for goal identification followed by the organization and classification of goals. Goal refinement involves the evolution of goals from the moment they are identified to the moment they are translated into operational requirements for the system specification [13].

## 2.5 NFR

The NFR framework focuses on the representation of non-functional requirements on the designed software system through interrelated goals. Three types of goals are defined: NFR, satisfying and argumentation goals. The last two model design decisions and arguments respectively. The NFR goals are soft goals which can be refined using different types of relationships describing how the satisfying of the offspring relates to the satisfying of the parent goal. A labeling procedure is defined for determining the degree of satisfying of each node in the goal structure [13].

## 2.6 MAP

A map is a diagram composed of nodes and edges. Nodes are *intentions* to realize and edges are strategies to reach these intentions. An edge enters a node if its associated strategy can be used for achieving the target intention. Since there can be multiple edges entering a node, a map is able for representing several ways to achieve an intention [13].

2.7 KAOS (Knowledge Acquisition in automated specification)

KAOS is a GORE method that raises the representation of a goals tree, which focuses on perform the process of formal analysis of requirements. The process for the mapped of KAOS goals diagram required that the secondary goals that subrogate the general goals is defined, then present them in goals more elementary than subrogate and so forth until you reach goals considered elementary or atomic or until expectations, requirements or domain properties [13].

Table I shows a set of activities for elaborating a KAOS model, and Table II shows the basic components of KAOS. There is a way to express a set of tasks and work products. This is called “activities”, and they define guidelines about how these tasks and work products should be used in practice [27].

Several authors have described some activities that KAOS uses to make its models [28]. These activities could be represented by SEMAT, using his activity spaces.

According to Figure 1, these activity spaces are focused on “Things to do” [27], in order to contemplate some important aspects such as the tasks of the model.

For each activity, there is an activity space where such an activity could be associated by using the relationships supported by SEMAT.

One of the relationships is “part of” and is used for joining activity spaces with activities. The other one is “Activity association” that is used for sequencing activities.

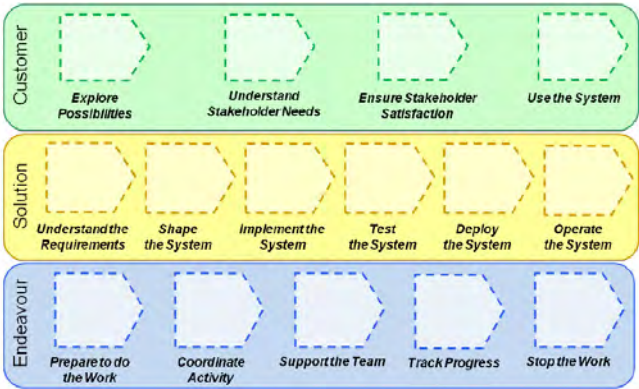


Figure 1. Activity spaces of the SEMAT essence [30].

**Table 1. activities for elaborating a KAOS model from [28].**

Activity	Sub-Activity	Description
Assign goals to agents		For each ASSUMPTION, the agent responsible is identified.
Goal elicitation	Asking HOW questions	Throught repeatedly asking how a goal is satisfied, new sub-GOALS are added to the initial set until the participants feel the level of detail described is adequate. This creates or extends the GOAL TREE.
	Asking WHY questions	Throught repeatedly asking why GOALS are included, new highlevel GOALS are added to the initial set until the underlying drivers for the system are discovered. This creates or extends the GOAL TREE.
	Defining objectives functions	One or more OBJECTIVE FUNCTIONS are created for each GOAL
	Defining quality attributes	One or more QUALITY ATTRIBUTES are created for each GOAL
Goal identification		The initial set of GOALS is created in discussion of all relevant stakeholders. GOALS may be preliminary at this state and subject to changes
Negotiating conflicts		Participants identify conflicting GOALS and negotiate which GOAL is more important and should be kept. The conflicting GOAL is removed.

**Table 2. Basic components of kaos.**

Component	Information
Object	Things of interest in the composite system whose instances may evolve from state to state. Objects can be entities, relationships, or events [17].
Operation	Input-output relations over objects. Operation applications define state transitions [17].
Agent	Kind of object that acts as a processor for operations. Agents are active components that can be humans, devices, software, etc. Agents perform operations that are allocated to them. KAOS lets analysts specify which objects are observable or controllable by agents [17].
Goal	It is defined as a “prescriptive statement of intent about some system whose satisfaction in general requires the cooperation of some of the agents forming that system”. Goals in KAOS may refer to services (functional goals) or to quality of services (non-functional goals). In KAOS, goals are organized in the usual AND/OR refinement abstraction hierarchies [17].
Requirement	A goal effectively assigned to a software agent [25].
Property domain	It is a descriptive assertion about object in the environment which holds independently of the system-to-be [25].
Expectation	A goal effectively assigned to an environment agent [25].

A collection of specifications in KAOS is a set of the following models: operating model, object model, and objective model. In general, each building on the KAOS language has two levels of structure: the outer graphical semantic layer where the concept is declared together with its attributes and relationships to other concepts, and the inner formal layer for formally defining the concept [17, 19, 22].

Figure 2 illustrates some basic notation for representing models by using KAOS.

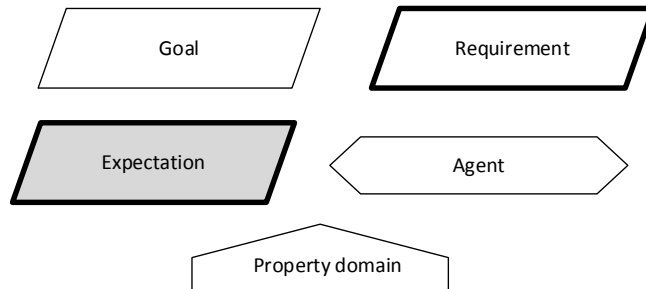


Figure 2. Basic notation for representing models by using KAOS from [24].

Building the goal diagram in KAOS is a task made by the analyst. This diagram is defined by interpreting the knowledge acquired about the area and the organization. Besides, this knowledge is based on the meetings with the stakeholders and the exploration about relevant documents [5].

The creation of a KAOS goal diagram exhibits some limitations: a set of verbs that can limit the definition of the different goals that represent the problem domain is absent. Hence, the analyst can use whatever verb he wants, and the diagram is conditioned by the analyst's subjectivity [21]. In order to overcome this limitation related to the subjectivity, Zapata *et al.* [24] propose the usage of pre-conceptual schemas, which allow for a closer approximation to the stakeholder discourse.

The structure of KAOS is represented hierarchically, connecting requirements and goals in graphical notation in an upward direction. The top goals are strategic objectives for the business, and the low side in the diagram reaches closer to the low level requirements [20].

## 2.8 SEMAT

Acronym of Software Engineering Method and Theory was founded by Ivar Jacobson, Bertrand Meyer, and Richard Soley, who felt the time had come to fundamentally change

the way people work with software-development methods. They wrote a call for action statement, which in a few lines identifies a number of critical problems, explains why there is a need to act, and suggests what needs to be done. The call for action is: Some areas of software engineering today suffer from immature practices. Specific problems include: the prevalence of fads more typical of the fashion industry than an engineering discipline; the lack of a sound, widely accepted theoretical basis; the huge number of methods and method variants, with differences little understood and artificially magnified; the lack of credible experimental evaluation and validation and the split between industry practice and academic research [13].

SEMAT identify a common ground for software engineering. This is manifested as a kernel of essential elements that are universal to all software-development efforts, and a simple language for describing methods and practices. Such kernel defines a set of Alphas representing essential things to work with (e.g., *Opportunity*, *Stakeholders*, *Requirements*, etc.) and a set of Space Activities representing the essential things to do (e.g., *Explore Possibilities*, *Understand Stakeholder needs*, *Understand the Requirements*, etc.). Methods, practices and the Essence Kernel itself are defined using the Essence Language [13].

The Kernel of SEMAT facilitates control over the activities performed during the software development process within which the following are noteworthy [13]:

**Stakeholders:** identified of stakeholders those are affecting the development of the software application; define the responsibilities of each stakeholder; the stakeholders are Authorizes to carry out their responsibilities within the team; the stakeholders define theirs expectative and needs to achieve with the software development.

**Opportunity:** Identified a commercial, social or business opportunity for the development of software to implement; determined a need a to be solved by a software application; the impact that will have software defined; the results are clear and quantified; defined that costs are less than the aggregate value that the software will bring for business.

**Requirements:** The purpose of the new system is established; define the requirements among the stakeholders and the team; establish and clarified the conflicts that may have the requirements; establish the priority requirements; agreed that the requirements are relevant with the business; the specified requirements reflecting what the system does and what does not do.

**Software System:** Identified the hardware platforms; selected the programming language and technologies; selected the software architect; defined boundary system; defined

an executable version for purpose and supports functional and non-functional testing; the system is operated by stakeholders; the functionality system is tested; the system is documented; the system is using in an operational environment.

Work: Identified that work has been requested; existing all conditions for starting the work; the progress of work is monitored; the completion of the work is verified; Work is officially ends.

Team: It has a clear mission for the team; it has a committed team; there is teamwork; works efficiently and effectively; controls in fulfilling the mission; produces Software System; performs and plans Work

Way of Working: principles and constraints are established; practices and tools are defined; method of work is used.

### 3. Integration into the semat kernel

This proposal is based on a previous work presented by Castro *et al.* [31]. SEMAT has a kernel that provides a common ground for methods and practices. This kernel has three features: it is operable, practical and extensible [14].

In the actionable portion of the kernel some alphas corresponding to the things “to work with” are defined: opportunity, stakeholder, requirements, system software, work, team and way of working.

The subset of alphas needed for describing the KAOS constructs is shown in Figure 3. Such a figure includes a set of universal alpha elements such as *Opportunity*, *Stakeholders*, and *Requirements*, and their relationships. These elements belong to the *Customer* and *Solution* areas of concern.

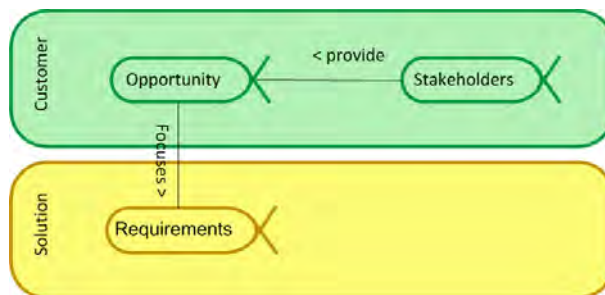


Figure 3. Subset of Alphas.



The next step is the inclusion of certain constructs such as elements of KAOS into the kernel relating them with the alphas.

These constructs or components of the metamodel have a relationship with certain alphas, which are related to the operable portion of the kernel [2]. Figure 4 shows the representation of the KAOS method by using the SEMAT kernel. The KAOS approach is mapped to the alphas of the kernel (i.e., placing *Goal* into Opportunity, *Agent* into Stakeholder, *Obtained goal-directed requirement* into Requirements, etc.).

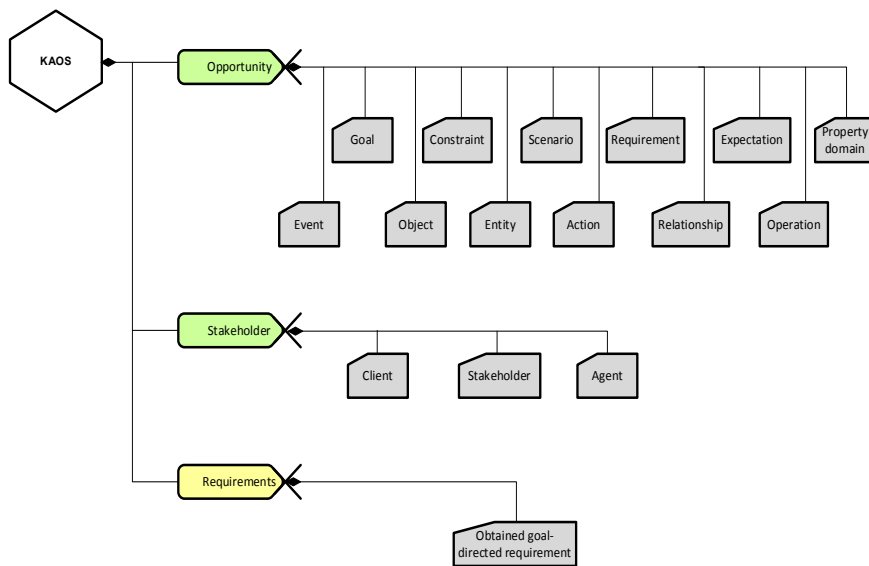


Figure 4. Alphas y Work Products representation.

As a result, we can perceive how these constructs included in SEMAT can be properly used for representing any model developed by KAOS. Figure 4 illustrates a case study by using the KAOS notation. This Figure shows the representation of a goal model by using KAOS. This Figure adapted from [23], presents a set of goals: *Effective Passengers Rapid Transportation*, *Safe Transportation*, *Train Progress*, *No Delay*, *No Train Collision*, *Doors Closed While Moving*, *Block Speed Limited*, *No Trains On Same Block*, and *Worst Case Stopping Distance Maintained*.

According to [26], the Essence language is defined as a set of constructs which are language elements defined in the context of a metamodeling framework. In this framework all the constructs of the language are at level 2.

- Level 3 – Meta-Language: the specification language, i.e., the different constructs used for expressing this specification, like “meta-class” and “binary directed relationship.”
- Level 2 – Construct: the language constructs, i.e., the different types of constructs expressed in this specification, like “Alpha”.
- Level 1 – Type: the specification elements, i.e., the elements expressed in specific kernels and practices, like “Requirements”.
- Level 0 – Occurrence: the run-time instances, i.e., these are the real-life elements in a running development effort.

An analyst using KAOS on a modeling project would be working at level 0. The goal set for each model would be a level 0 instance of the “Goal” work product defined at level 1. This is exactly analogous to the creation of Goals “*Effective Passengers Rapid Transportation*”, “*Rapid Transportation*”, “*Safe Transportation*”, etc.

In order to define the dynamic semantics, it is necessary to refer to the inhabitants of levels 1 and 0 as well as those of level 2. In order to make it clear at which level a named term belongs, Essence uses the following naming convention [26]:

- X (an unadorned name) is a language Construct at level 2, such as Work Product.
- my\_X (prefixed) is a Type at level 1 created by instantiating X. So if X is Work Product, my\_WorkProduct could be “Goal.”
- my\_X\_instance is an Occurrence at level 0 by instantiating my\_X. So if X is Work Product, my\_WorkProduct\_instance could be the specific goal related to *Rapid Transportation*.

Essence explicitly defines Domain classes, such as *my\_WorkProduct*, that contains the necessary instance properties (defined as *EndeavourProperty* instances from the meta-model), that is to be endowed at enactment [26]. As can be seen, in Figure 9 by adding the *Goal* instance to the class *my\_WorkProduct* we can support the construct *Goal* as defined in the KAOS specification.

So, each of goals in Figure 7 can be mapped in the SEMAT kernel as illustrated in Figure 9.

Comparing the KAOS specification and the Essence approach shows that both can be supported with similar foundation. In this case, the Essence approach is separated into

a metamodel (i.e., writing system or language as understood in the OMG context) and the Kernel providing the common starting ground.

Consequently, the KAOS goal instances illustrated in Figure 7 map to concepts in the Essence Language as demonstrated in Figure 9.

In addition, the activities that are used by KAOS, could be represented in SEMAT activity spaces as it is shown in Figure 5.

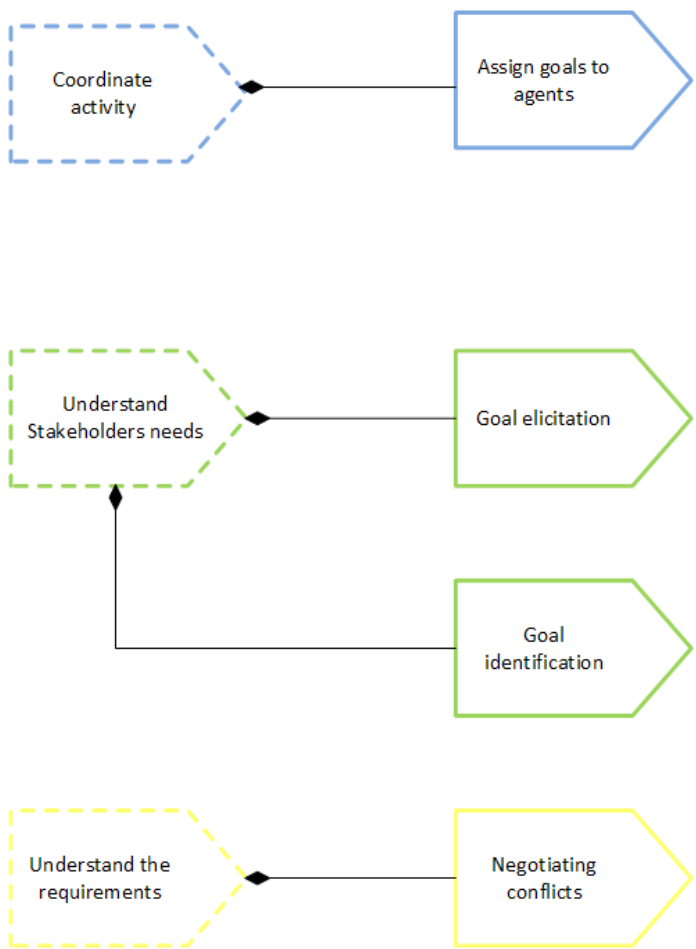


Figure 5. Activities in KAOS.

KAOS activities have a sequence, and this sequence also could be represented by SEMAT as it is presented in Figure 6.

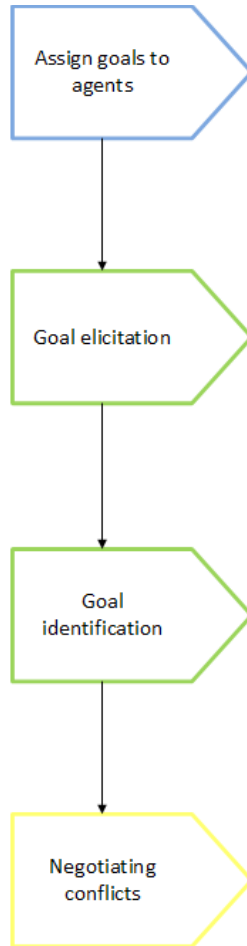


Figure 6. Sequence of KAOS activities from [27].

Each element of the model showed in Figure 5 can be represented by using the Semat Kernel. This representation is illustrated in Figure 8. As we can see the basic elements have been modeled. In addition, we use the *role* element in order to represent the *Analyst*, who is charged with the task of elaborating the model. This role works on two work products: Goal and Requirement. Such artifacts are associated with four activities: Assign goals to agents, Goal elicitation, Goal identification, and negotiating conflicts. These activities are grouped in several activity spaces as follows: *Assign goals to agents* grouped by *Coordinate activity*, *Goal elicitation* and *Goal identification* grouped by *Understand stakeholder needs*, and *negotiating conflicts* grouped by *Understand requirements*.

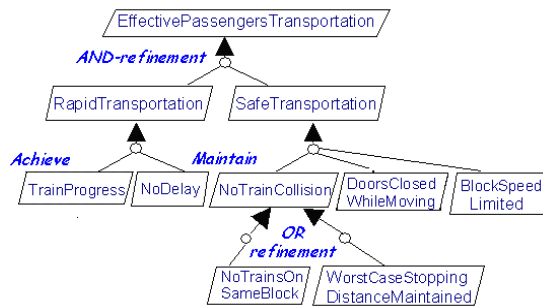


Figure 7. KAOS diagram defined by the Stakeholder adapted from [23].

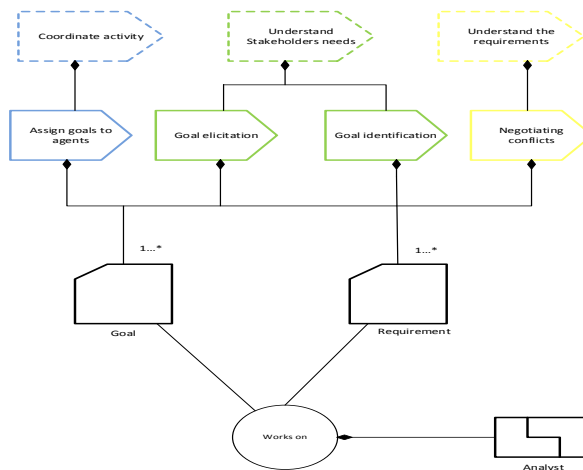


Figure 8. Integrated representation of a KAOS model by using the Semat Kernel.

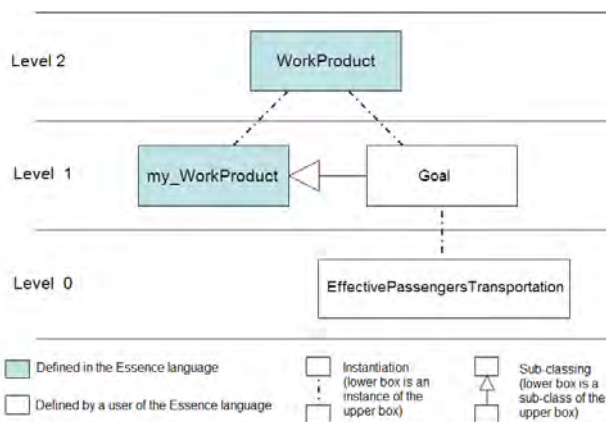


Figure 9. KAOS goal represented by SEMAT based on [26]

## 4. Conclusions and future work

In this chapter we propose a representation of KAOS by using the SEMAT kernel. We face some issues studied by [2], which are needed for achieving an adequate understanding of their metamodel.

This work establishes a common ground for studying and integrating various methods and practices. In order to achieve this, the proposed approach allows us map the elements into the elements used by the SEMAT kernel.

As future work, we can use the KAOS representation obtained for leading new works which facilitate the understanding of the different GORE methods founded in the literature. Besides, other works fostering the inclusion of such methods in the industrial context are required.

## 5. References

- [1] I. J. Jureta, and S. Faulkner, "Clarifying Goal Models," In Proc. Tutorials, posters, panels and industrial contributions at the 26th International Conference on Conceptual Modeling, Auckland, New Zealand, 2007, pp. 139–144.
- [2] A. Dardenne, A. Van Lamsweerde, and S. Fickas, "Goal-directed requirements acquisition," *Science of computer programming*, vol. 20(1), 1993, pp. 3–50.
- [3] R. Matulevičius, P. Heymans, and A. L. Opdahl, "Comparing GRL and KAOS using the UEML Approach," *Enterprise Interoperability II. New Challenges and Approaches*, 2007, pp. 77–88.
- [4] P. Espada, M. Goulão, and A. João, "A framework to evaluate complexity and completeness of KAOS goal models," in *CAiSE'13 Proceedings of the 25th international conference on Advanced Information Systems Engineering*, Berlin: Springer, 2013, pp. 562–577.
- [5] F. Vargas, and C. Zapata, "Modelo para la Especificación de requisitos iniciales de software a partir de la relación sintáctica y semántica entre objetivos y problemas," *Facultad de Minas Universidad Nacional de Colombia Sede Medellín*, 2015.
- [6] A. Babar, B. Wong, and A. Q. Gill, "An evaluation of the goal-oriented approaches for modelling strategic alignment concept," In *2011 Fifth International Conference on Research Challenges in Information Science*, Guadeloupe - French West Indies, France, 2011, pp. 1–8.
- [7] B. González-Baixaui, M. A. Laguna, and J. C. do Prado Leite, "Análisis de Variabilidad con Modelos de Objetivos," In *WER*, 2004, pp. 77–87.
- [8] E. Kavakli, "Goal-oriented requirements engineering: A unifying framework," *Requirements Engineering*, vol. 6(4), 2002, p.p. 237–251.

- [9] V. M. B. Werneck, A. D. P. A. Oliveira, and J. C. S. do Prado Leite, "Comparing GORE Frameworks: i-star and KAOS," In WER, 2009.
- [10] G. Koliadis, and A. Ghose, "Relating business process models to goal-oriented requirements models in KAOS," In Advances in knowledge acquisition and management, Springer Berlin Heidelberg, 2006, pp. 25-39.
- [11] ITC-irst, and U. d. Trento, "The Tropos Methodology and its Metamodel," 2006.
- [12] R. Darimont, and M. Lemoine, "Goal-oriented Analysis of Regulations," In ReMo2V, 2006.
- [13] C. M. Zapata, L. F. Castro, F. A. Vargas, "GBRAM from a SEMAT Perspective," Methods, modeling, and teaching, 2014, vol. 3, pp. 21-26.
- [14] I. Jacobson, P. W. Ng, P. E. McMahon, I. Spence, and S. Lidman, The essence of software Engineering: applying the SEMAT kernel, Addison-Wesley, 2013.
- [15] M. J. Escalona, and N. Koch, Ingeniería de Requisitos en Aplicaciones para la Web—Un estudio comparativo, Universidad de Sevilla, 2002.
- [16] M.A. Laguna, and B. González-Baixauli, "Product Line Requirements: Multi-Paradigm Variability Models," In WER, 2008.
- [17] A. Lapouchnian, Goal-Oriented Requirements Engineering, Toronto, 2005.
- [18] E. Yu, Modelling Strategic Relationships for Process Reengineering, Ph.D. Thesis, University of Toronto, Toronto, 1995.
- [19] J. Guzman, L. A. Lezcano and S. A. Gómez, "Characterization of the elements the goal diagram KAOS from natural language," revista Colombiana de tecnologías avanzadas, vol. 1 (21), 2012, pp. 138-144.
- [20] F. Almisned, and J. Keppensm, "Requirements Analysis: Evaluating KAOS Models," Journal of Software Engineering and Applications, vol. 3 (9), 2010, pp. 869-874.
- [21] C. Zapata, and L.A. Lezcano, "Characterization of goal diagram verbs", *Revista Dyna*, vol. 72 (158), 2009, pp. 219-228.
- [22] A. van Lamsweerde, Goal-oriented requirements engineering: A guided tour, 2001.
- [23] C. Ponsard, P. Massonet, A. Rifaut, J.F. Molderez, A. van Lamsweerde and H. Tran Van, "Early verification of mission critical systems," Electronic Notes in Theoretical Computer Science, vol. 33, 2005, pp. 237-254.
- [24] C.M. Zapata, L.A. Lezcano, and P.A. Tamayo, "Preconceptual-schema-based representation of KAOS goal diagram," In Computing Congress (CCC), 2011 6th Colombian, 2011, pp.1-6.
- [25] R. Matulevičius, and P. Heymans, Analysis of KAOS Meta-model. Namur University Computer Science Department, Belgium, 2005
- [26] OMG, "Essence - Kernel and Language for Software Engineering Methods, Initial Submission - Version 1.1," Object Management Group (OMG), OMG Document ad/15-12-02, December 2015.
- [27] C.M. Zapata, "Cosas que siempre hacemos: Los espacios de actividad", Junio 2016.

- [28] J. Buis, "GORE: Goal Oriented Requirements Engineering," marzo 2014.
- [29] L. Duboc, E. Letier, D.S. Rosenblum, and T.A. Wicks, "A case study in eliciting scalability requirements," *International Requirements Engineering, RE'08*. 16th IEEE, 2008, pp. 247-252.
- [30] C.M. Zapata, M.D. Rojas, R.E Arango, and L.D. Jiménez, "SEMAT GAME: Applying a project management practice," 2015.
- [31] L.F. Castro, E. Espitia, and S. Montaña, "Goal Oriented Requirements Engineering supported by the SEMAT kernel," *CONISOFT 2016*, Puebla, Mexico, 2016.



# Chapter # 19

## QUACOP: An approach to Increase the Quality of Artifacts considered in a Project Planning Process

**César Guerra-García,  
Rafael Llamas, Omar  
Montaño**

Department of  
Information Technology  
Polytechnic University  
of SLP  
San Luis Potosí, México  
{cesar.guerra, rafael.  
llamas, omar.montano}@  
upslp.edu.mx

**Reyes Juárez-Ramírez**

School of Chemical  
Science and Engineering  
Autonomous University  
of Baja California  
Tijuana, B.C., México  
reyesjua@uabc.edu.mx

**Victor Menéndez  
-Domínguez**

Department of  
Mathematics  
Autonomous University  
of Yucatán  
Mérida, Yucatán, México  
mdoming@uady.m

### 1. Introduction

At present, both public and private organizations understand and assess the value of data. Well managed enterprise information has inestimable value for the organizations and enterprises [1]. Once the data is considered as fundamental asset for organizations, its strategic value leads to reconsider the importance of maintaining adequate levels of quality in data that is managed by any kind of enterprise application.

The data are becoming a key asset to improve the efficiency in today's dynamic and competitive business environment [2]. The economic and social impact of poor data quality (DQ) has a significant economic cost for organizations [3]. Although significant works of research have been done on the notion of data and information quality, their main focus is on the objective quality attributes of data [4]. The quality of data products and data warehouse directly determines the quality of the business operations management and decision-making through data [5]. Given this strategic value of data in the execution of business processes, and considering that more frequently organizational data is published through Web applications, organizations need to ensuring acceptable levels of quality.

For many years the data quality has been treated as a set of measurements related to a specific set of data (e.g. ¿are they complete? ¿are they valid? or at the intersection of two sets of data ¿are they consistent?). When we discuss about data quality, it is thought that quality is based on the data stored in the storage repositories (e.g. databases, text files, excel files, etc.). This idea is very biased, as mentioned in Strong et al. in [6] the concept of “quality goes beyond data stored, it includes the data found in the processes of production and its use”.

In the last years the research about diverse topics of data quality has been developed rapidly, becoming a popular field of study, which covers areas like information systems, knowledge management, artificial intelligence, accounting, statistics and other disciplines [7]. One of the approaches through which it seeks to improve the quality of software, is focusing on the study and improvement of the processes that govern the development of software [8], and it is precisely the core of our research in this case.

Improve software quality involves not only the creation of effective programming languages, new development methodologies (e.g. agile, XP) and advanced tools (IDE's, compilers, debuggers, etc...), and focusing only on the final quality of the software product. It also includes the procedures used to create, deliver and maintain software. Fuggeta in [8] gives a definition of Software Process to affirm that “A coherent set of policies, organizational structures, technologies, procedures and artifacts, which are necessary to design, develop, deploy and maintain a software product”. The use of software development processes adjusted to the requirements of a project and of the work team has strong influence in the final quality of the produced products.

As can be observed into a Software Development Project, there are a lot activities that are executed, different artifacts are generated and used, many data are interchanged among different actors and output products are generated. This outline is complex and is explained by the participation of several processes found within it, as it is shown by the standard ISO/IEC 12207:2008 [9] into the group of “Project Processes” of the software life cycle.

Within software development project, the quality of the artifacts is related to the data quality and its content, besides also could be evaluated through its structure and metadata. But being careful not to misinterpret the quality of the artifacts with the quality of the data used in the artifacts, nor with the quality of the data describing artifacts.

One of the problems with which the management of software development projects is faced is the misuse of resources. And one of its causes is originated by improper

decisions leading by an inadequate level of the quality of an artifact. This low level of quality can also be caused by poor presentation and/or description of the artifact (through its metadata). This is a problem because once we are not able to define criteria of utility of the artifact, this one could be ignored or if it is considered it could generate errors in some scenarios (e.g. lack of update). The quality models can be used to evaluate the final product or the different artifacts produced along with the software development.

In this paper, we focus on studying the data quality corresponding to the metadata associated with the artifacts that describe and provide additional information (e.g. title, ID, version, update, etc.) on them to managers of software development projects.

The main goal of this paper is show a Data Quality Model that could be used by Software Team Leaders and Project Managers, as a reference to evaluate, and if appropriate, to improve the level of quality of the values corresponding to the metadata that describes the artifacts used in the planning process of a software development project.

In order to propose this model of data quality, we take into account as reference, the artifacts used in the “Planning Process”, defined in the international standard ISO/IEC 12207:2008 [9], which can be modified by the development group as deemed for each project.

The identification of the specific structure of artifacts will be derived according to PMBOK (Project Management Body of Knowledge) [10]. Initially, DQ dimensions are taken from the de facto standard provided by Strong et al. in [11], being the definitions of some of these dimensions suggested by Pipino et al. in [12].

This proposal will be used because it is more generic than that provided by the international standard ISO/IEC 25012 [13], which focuses exclusively on the dimensions required by the structured data within a computer system.

This work is organized as follow: in section 2 we describe the proposed models by Strong et al. [6] and Pipino et al. [12], both respected to the data quality dimensions, besides the international standard ISO/IEC 12207:2008 [9] and the PMBOK [10]. In section 3, the artifacts of the Planning Process are identified and the metadata are proposed. Section 4 shows the Data Quality Model applicable to the metadata. Finally, in section 5 the conclusions are shown.

## 2. Related Areas

### 2.1 Models of Data Quality

In order to minimize a negative impact of problems due to low levels of data quality, it is big paramount that companies can have a quantitative perception of their importance.

The organizations and companies must assess how good their organizational data resources are for the tasks at hand. This is a challenge that still today stay vigent. Organizations have to deal to the data quality issues, both in subjective manner by individuals that are using the data, as objective manner, with measures based on a set of data.

Moreover, different data quality dimensions have been defined by several authors (from a different point of view and use) and even they have been defined in the international standard ISO 25012. This remarks the importance that data quality topic has been gaining in the last two decades, and to perform their definition influences the context in which it is to be used.

In this proposal as we said before, two data quality models are used as basis; by one hand, the DQ Model proposed by Strong et al. [6], the authors show four categories and describe on each one of them some dimensions (see Table 1).

On the other hand, the model of data quality proposed by Pipino et al. [12], Pipino describes in deep sixteen data quality dimensions, they are showed in Table 2.

**Table 1.** Data quality model proposed by strong et al. [6]

Category of data quality	Dimensions of data quality
Intrinsic	Accuracy, Objectivity, Believability, Reputation
Accessibility	Accessibility, Access security
Contextual	Relevancy, Value-added, Timeliness, Completeness, Amount of data
Representational	Interpretability, Ease of understanding, Concise representation, Consistent representation

Previous to the elaboration of the model, we carried out modifications in the definitions of some data quality dimensions, in order to adequate them to the context of metadata of the artifacts. These modifications were focused on the quality of the artifacts, through the metadata that describes them, besides to propose additional information of each them (e.g. title, version, updated, etc.).

**Table 2. Data quality model proposed by pipino et al. [12]**

Dimension	Definition
Accessibility	Data is available or easily and quickly retrievable.
Appropriate Amount of data	Volume of data is appropriate for the task at hand.
Believability	Data is regarded as true and credible.
Completeness	Data is not missing and is of sufficient breadth and depth for the task at hand.
Concise representation	Data is compactly represented.
Consistent representation	Data is presented in the same format.
Ease of manipulation	Data is Easy to manipulate and apply to different tasks.
Free-of-error	Data is correct and reliable.
Interpretability	Data is in appropriate languages, symbols, and units, and the definitions are clear.
Objectivity	Data is unbiased, unprejudiced, and impartial.
Relevancy	Data is applicable and helpful for the task at hand.
Reputation	Data is highly regarded in terms of its source and content.
Security	Access to data is restricted appropriately to maintain its security.
Timeliness	Data is sufficiently up-to-date for the task at hand.
Understandability	Data is easily comprehended.
Value-added	Data is beneficial and provides advantages from its use.

Once established the definitions of the data quality dimensions that are going to serve as a reference, the next step is to revise the approaches about the processes that are involved in a software project.

## 2.2 Software Life Cycle Process according to Standard ISO/IEC 12207:2008

According to the international standard ISO/IEC 12207:2008 [9], the Software life cycle contains 43 processes (see Figure 1). Each of these processes uses as input products different specific artifacts, and generates others as output products. Among all of these, this chapter is focused on “Project Planning” process, because through it the entire project management is planned and started.

System Context Processes			Software Specific Processes	
Agreement Processes	Project Processes	Technical Processes	Software Implementation Processes	Software Support Processes
Acquisition Process	Project Planning Process	Stakeholder Requirements Definition Process	Software Implementation Process	Software Documentation Management Process
Supply Process	Project Assessment and Control Process	System Requirements Analysis Process	Software Requirements Analysis Process	Software Configuration Management Process
	Decision Management Process	System Architectural Design Process	Software Architectural Design Process	Software Quality Assurance Process
	Risk Management Process	Implementation Process	Software Detailed Design Process	Software Verification Process
	Configuration Management Process	System Integration Process	Software Construction Process	Software Validation Process
	Information Management Process	System Qualification Testing Process	Software Integration Process	Software Review Process
	Measurement Process	Software Installation Process	Software Qualification Testing Process	Software Audit Process
		Software Acceptance Support Process		Software Problem resolution Process
		Software Operation Process	Software Reuse Processes	
		Software Maintenance Process	Domain Engineering Process	Reuse Program Management Process
		Software Disposal Process	Reuse Asset Management Process	

Figure 1. Project processes defined by ISO/IEC 12207:2008.

The standard ISO/IEC 12207:2008 established that the process of “Project Planning” has as purpose: “to produce and communicate effective and workable project plans”.

The Project Planning process determines “the scope of the project and technical activities, identifies process outputs, project tasks and deliverables, establishes schedules for project task conduct, including achievement criteria, and required resources to accomplish project tasks” [9].

## 2.3 Processes Groups according to PMBOK

Another approach where the processes involved in a project are defined is the one with the Project Management of Body of Knowledge. The project management is the application of knowledge, skills, techniques and tools to project activities to meet project requirements [10]. The project management is accomplished through the application of many processes as: initiating, planning, executing, monitoring, controlling and closing. In this approach, the artifacts are defined as input and output products.

The PMBOK is organized in nine knowledge areas and a collection of processes; all them accepted as best practices in the organizations around the world and becoming in

a standard de facto in the Project Management discipline [10]. PMBOK defines the project as “a temporary effort undertaken to create a product, service or result”; it identifies 44 processes organized in these knowledge areas: project integration, project scope, project time, project cost, project quality, project human resources, project communications, project risk and project procurement (see Figura 2).

Each process covers some particular products (both input and output), besides of specific tools and techniques. These products are similar to the artifacts used in the standard ISO/IEC 12207:2008.

That said, it is possible to propose equivalence between both standards, with respect to the artifacts that are utilized in the Planning Process of a Software Development Project. This step is justified because through the PMBOK could be known both the name of the artifact as the structure that should be fulfilled. As mention before, in this proposal the inputs and outputs products defined in the PMBOK are taken as base.

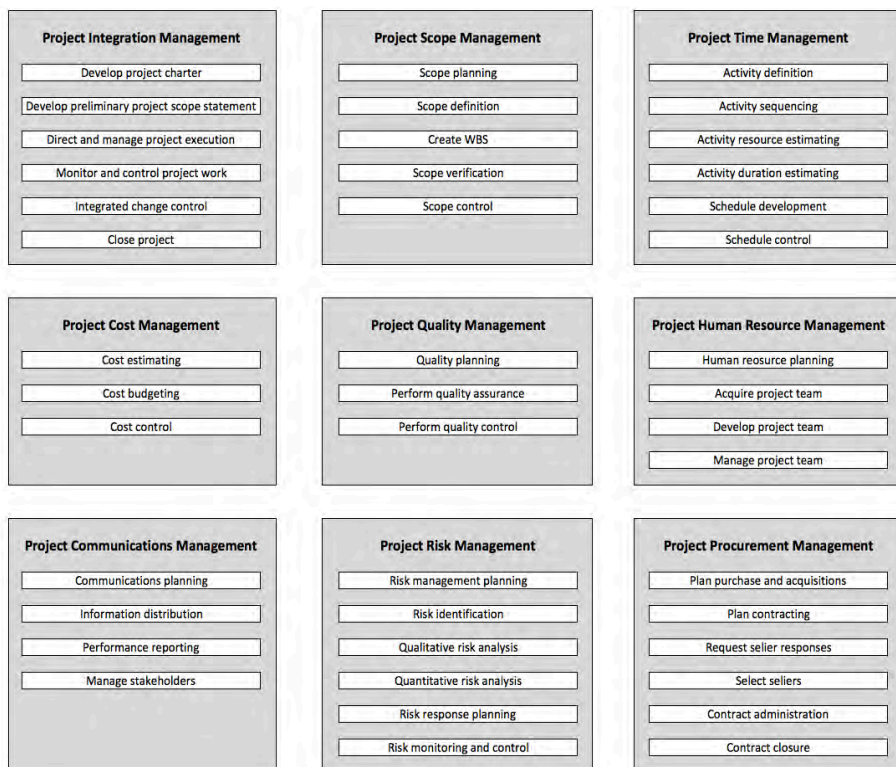


Figure 2. Project management knowledge areas and processes.

### 3. Identification of artifacts of the project planning process versus pmbok products

The proposal begins with the analysis of the all artifacts that participate in the Project Planning process, which could be modified with discretion by the development team members.

Later on, a set of metadata are identified whose main function is, one the one hand, to describe each artifact; and on the other hand, to define an internal content structure.

With respect to the structure that each artifact should present (it is oriented to its content), we propose to use as basis the PMBOK structure. This structure could be considered as specific metadata to each artifact and oriented them to its content. Therefore, on these metadata of content, you can also evaluate and propose a data quality model (this DQ model is explained in deep in section 4).

Thus, subsequent questions are twofold: first solve ¿what are the artifacts? and later ¿what are the metadata necessary to describe whatever artifact?.

For the identification of the artifacts involved in the Project Planning process, we conduct a revision as indicated by standard ISO/IEC 12207:2008, and equivalence was done as indicated by the PMBOK (respect to the input/output products of the processes in a project).

As a first result of this analysis the next artifacts were identified (see Table 3 to Table 9).

**Table 3. Proposal of equivalence between artifacts ISO/IEC 12207:2008 and the input/output products of PMBOK “project scope statement”**

Artifacts by ISO 12207:2008 Project Planning Process	Input/Output products defined by PMBOK	
	Name	Structure (Metadata of content for each artifact)
The scope of the work for the project is defined. The feasibility of achieving the goals of the project with available resources and constraints are evaluated. Interfaces between elements in project, and with other project and organizational units, are identified.	Project scope statement.	Project scope. Scope planning. Deliverables of project. The hypothesis of the project. Thr Project restrictions. Description of work.



**Table 4. Proposal of equivalence between artifacts ISO/IEC 12207:2008 and the input/output products of PMBOK “dictionary of work breakdown structure”**

Artifacts by ISO 12207:2008 Project Planning Process	Input/Output products defined by PMBOK	
	Name	Structure (Metadata of content for each artifact)
<p>The tasks and resources necessary to complete the work are sized and estimated.</p> <p>The Manager shall establish the requirements for the project to be undertaken.</p> <p>Estimation of effort.</p> <p>Adequate resources needed to execute the tasks.</p> <p>Allocation of tasks.</p> <p>Assignment of responsibilities.</p>	Dictionary of Work Breakdown Structure (WBS).	<p>Description of each component in the WBS.</p> <p>For each component of WBS is included a brief definition of the scope or work statement, deliverables, a list of activities and milestone associated.</p> <p>Could include:</p> <ul style="list-style-type: none"> <li>Responsible organization.</li> <li>Dates of starting and ending.</li> <li>Resources required.</li> <li>Cost estimating.</li> <li>Contract information.</li> <li>Requirements of quality.</li> <li>Technical references to facilitate performance of the work.</li> </ul>

**Table 5. Proposal of equivalence between artifacts ISO/IEC 12207:2008 and the input/output products of PMBOK “project management plan”**

Artifacts by ISO 12207:2008 Project Planning Process	Input/Output products defined by PMBOK	
	Name	Structure (Metadata of content for each artifact)
<p>Plans for the execution of the project are developed.</p> <p>Plans for the execution of the project are activated.</p>	Project Management Plan.	<p>Description how to execute, monitor and control the project.</p> <p>Could include:</p> <ul style="list-style-type: none"> <li>One or more subsidiary management plans.</li> <li>Planning documents.</li> <li>Organizational process assets.</li> <li>Communications management plan.</li> </ul>

**Table 6. Proposal of equivalence between artifacts ISO/IEC 12207:2008 and the input/output products of PMBOK “project schedule”**

Artifacts by ISO 12207:2008 Project Planning Process	Input/Output products defined by PMBOK	
	Name	Structure (Metadata of content for each artifact)
<p>Schedules for the timely completion of tasks.</p>	Project Schedule.	<p>Activity definition.</p> <p>Activity duration estimating.</p> <p>Activity sequencing.</p> <p>Dates to achieve the planned schedule milestones (schedule development).</p> <p>Activity resource estimating.</p> <p>Staffing management.</p>

**Table 7. Proposal of equivalence between artifacts ISO/IEC 12207:2008 and the input/output products of PMBOK “risk register”**

Artifacts by ISO 12207:2008 Project Planning Process	Input/Output products defined by PMBOK	
	Name	Structure (Metadata of content for each artifact)
Quantification of risks associated with the task or the process itself.	Risk register.	Qualitative risk analysis. Quantitative risk analysis. Risk response planning. Could include: Risk description. Category. Cause. Occurrence probability. Impact in the objectives. Proposal of responses. The person in charge. Actual condition.

**Table 8. Proposal of equivalence between artifacts ISO/IEC 12207:2008 and the input/output products of PMBOK “quality management plan”**

Artifacts by ISO 12207:2008 Project Planning Process	Input/Output products defined by PMBOK	
	Name	Structure (Metadata of content for each artifact)
Quality assurance measures to be employed throughout the project.	Quality management plan.	Description of quality policies of the performing organization. Description of the processes of quality planning, assurance and control quality. Description of quality metrics. Description of the quality checklist. Definition of quality baseline.

**Table 9. Proposal of equivalence between artifacts ISO/IEC 12207:2008 and the input/output products of PMBOK “cost management plan”**

Artifacts by ISO 12207:2008 Project Planning Process	Input/Output products defined by PMBOK	
	Name	Structure (Metadata of content for each artifact)
Costs associated with the process execution.	Cost management plan.	Accuracy level of cost estimates. Including contingency costs. Measure units (hours, days, weeks, etc.). Links to the procedures of the organization (with respect to the accounting of the organization). Rules for the measuring of performing. Formats of cost reports and its presentation frequency. Description of cost processes: estimating, budgeting and control. Contract statement of work.

### 3.1 Proposal of Metadata for artifacts

Once analyzed all the artifacts selected, we show a set of metadata proposed (all of them are presented in Table 10). It is worth to highlight that these metadata are common to all artifacts identified (being necessary to identify others specific metadata for each one of artifacts in particular, which could be one of the future work of the research).

Table 10. Set of metadata identified for all artifacts of the project planning process

<ul style="list-style-type: none"><li>• Title of document.</li><li>• ID of the document.</li><li>• Table of contents.</li><li>• Revision history, including:<ul style="list-style-type: none"><li>» Date.</li><li>» Version number.</li><li>» Description of revision.</li><li>» Author.</li><li>» Reviser.</li></ul></li><li>• Change request number.</li><li>• Organization or department in charge to elaborate the document.</li><li>• Indicate if the document is “Confidential” and/or the access level to it.</li><li>• Objective of developing the artifact.</li><li>• Glossary.</li></ul>
--

Once identified the artifacts, the metadata associated and taking as reference the definitions of the data quality dimensions (Table 2), in the next section we show the Data Quality Model that is applied to the metadata aforementioned.

## 4. Proposal of data quality model applied to the metadata of the artifacts

The data quality models aforementioned (Table 1 and 2) were considered as a reference in order to elaborate our proposal.

This data quality model proposed should be applied to all metadata of the artifacts identified in Table 10. It is worth to highlight that the experience of the authors in many software development projects were considered, in order to complement and improve the proposal.

Each one of the metadata will be assessed from the point of view of the next dimensions shown in Table 11. In order to get a better understanding, a brief description is given for each dimension.

**Table 11. Data quality model suggested to be applied to the metadata of the artifacts in a project planning process**

Dimension	Description
Accessibility	The metadata "Title" and "Revisions history", must have a reliable level and should contain a procedure that ensures its credibility.
Beliavability	The artifacts must have a reliable "Title". The "Revisions history" should follow a procedure that ensures its credibility.
Timeliness	The "Revisions history" should show that is updated and it has been revised to maintain its effective time. In case that document had been modified the "Table of contents" should be updated accordingly.
Security	The "Title of document" should be public, however, it is necessary to indicate which others metadata could be publics or confidential. For instance, the metadata "Person in charge of the document" could be public or not, depending of the scenario.
Reputation	The procedure to assign the "Revisions history" should ensure a tracking update that has seen the artifact.
Interpretability	The "Objective of developing the artifact" should be correctly expressed, by using the adequate language and symbols. A particular format should be used to assign the "ID of the document" through the specific language.
Objectivity	The "Title of document" and "Objective of Developing the artifact" should be unbiased, according to the content of the artifact.
Relevancy	It is imperative to determine the level of "Confidentiality" as well as the "Organization or Department in charge of document".
Understandability	The "Glossary" should be easily realized, in order to get a better understanding of all artifacts.
Consistent representation	Each element into "Revisions history" has to show the same presentation format.
Completeness	The "Objective of developing the artifact" must be completed, in order to clearly describe the objective. The "Table of contents" should be complete, thus assuring that the document contains all the information. The "ID of document" always must be complete.
Accuracy	Both the "Title of document" as "Version number" should be accurate. The "Revisions history" should be precise avoiding data show more. The "ID of document" should be accurate.
Free of error	The "Title of the document" and "Table of contents" should be shown without any kind of error.

## 5. Conclusions

During last two decades, the variety and complexity of software projects has grown dramatically, satisfying any kind of business processes into the companies. The software in general and particular Web applications are as important as organization itself.

This paper shows a particular data quality model applied to a set of metadata that describe the artifacts used in a project planning process. This data quality model focuses on the quality of the artifacts through their metadata that describe them, rather than the quality of the data used in the content of the artifacts.

The strength and advantage of this data quality model is based once that it is correctly applied in a software development project, the artifacts are become in active elements, reference and support to the management of the project managers.

Likewise, can be observed that artifacts defined in the international standard ISO 12207:2008 are also defined into the PMBOK, so we can affirm that both standards can complement.

Until our knowledge, there is not other proposal related to this topic, except the proposal presented in [14], being this work a new extended and improved version.

As future work, we will continue with the analysis of the rest of processes defined in the standard ISO/IEC 12207:2008, trying to identify their artifacts and comparing them with those defined in PMBOK; likewise, identifying metadata by each artifact. The above as a preamble to define metrics, and later on to evaluate software projects based on these.

## 6. References

- [1] George, E. and J. Gao. A quality study exploring users' perception of information management challenges in the cloud. in International Conference on Information Quality, ICIQ 2014. 2014. Xian, China.
- [2] Oliveira, P., Ft. Rodrigues, and P. Henriques. A formal Definition of Data Quality Problems. in Tenth International Conference on Information Quality (ICIQ'05). 2005. MIT, Cambridge, MA, USA.
- [3] Eppler, M. and M. Helfert. A Classification and Analysis of Data Quality Costs. in International Conference on Information Quality. 2004. MIT, Cambridge, MA, USA.
- [4] Eshraghian, F. and S.A. Harwood. Information product: How information consumers' perception of 'fitness for use' can be affected. in International Conference on Information Quality, ICIQ 2015. 2015.
- [5] Liu, Q., G. Feng, and N. Wang. Managing data quality for information systems combination from different data sources in International Conference on Information Quality, ICIQ 2014. 2014. Xian, China.
- [6] Strong, D.M., Y.W. Lee, and R.Y. Wang, Data Quality in Context. Communications of the ACM, 1997. 40(5): p. 103-110.

- [7] Bai, X., M. Nunez, and J.R. Kalagnanam, Managing data quality risk in accounting information systems. *Information Systems Research*, 2012. 23: p. 453-473.
- [8] Fuggeta, A. Software Process: A Road Map. in *Twenty-Second International Conference on Software Engineering (ICSE'2000)*. 2000. Limerick, Ireland: ACM Press.
- [9] ISO/IEC, ISO/IEC 12207. International Standard. Software Life Cycle Processes., 1995, International Standard Organization/International Electrotechnical Committee: Geneve.
- [10] PMI, A Guide to the Project Management Body of Knowledge, 2000 edition. Project Management Institute Communications, United States, 2000.
- [11] Strong, D., Y. Lee, and R. Wang, Ten Potholes in the Road to Information Quality. *IEEE Computer*, 1997: p. 38-46.
- [12] Pipino, L., Y. Lee, and R. Wang, Data Quality Assessment. *Communications of the ACM*, 2002. 45(4): p. 211-218.
- [13] ISO-25012, ISO/IEC 25012: Software Engineering-Software Product Quality Requirements and Evaluation (SQuaRE)-Data Quality Model, 2008.
- [14] Guerra-García, C., et al., Improving the Project Planning Process Considering Artifacts with Quality, in *4th. International Conference on Software Engineering Research and Innovation*, IEEE, Editor 2016, IEEE: Puebla, Pue. p. 15-20.

# Chapter # 20

## New relationships of the Risk alpha with the Semat Essence kernel

**Carlos Mario Zapata Jaramillo & Antony de Jesús Henao Roqueme**

Departamento de Ciencias de la Computación y de la Decisión

Universidad Nacional de Colombia—Sede Medellín

Medellín, Colombia

cmzapata@unal.edu.co , ajhenaor@unal.edu.co

### 1. Introduction

The lack of a common ground for comparing software engineering methods and practices makes difficult the organizational selection of the appropriate set of practices for guiding the work. As a result, a framework for supplying the lack of a common ground is needed [1].

Semat (Software Engineering Method and Theory) is a proposal intended to supply the need for a common ground in the software engineering discipline. As a result, Semat defines a theory composed of universal elements covering all software engineering endeavors [2].

Alphas are part of the universal elements defined by Semat and they represent “the things we always work with.” Alphas are the most important dimensions we have to care about in a software engineering endeavor. Semat allows for assessing health and progress of a software engineering endeavor via alpha states. An alpha state is defined by using a checklist with the criteria the team should fulfill for achieving such state. We can review how a software engineering endeavor is advancing by recognizing the state an alpha has. Hence, alpha states lead to the Essence kernel actionability [1].

Essence kernel excludes risk as an alpha. However, according to Jacobson et al. [3] the risk alpha was neglected as an alpha for the Essence kernel, since they consider checklists in the Essence kernel already address risks threatening software engineering endeavors.

According to Zapata and Henao [4], excluding the risk alpha from the Essence kernel jeopardizes the actionable feature of the Essence kernel. Also, checklists in the Essence kernel addressing risks left unguided the things have to be done for achieving some alpha states, which runs counter to checklist definition.

Similar to the Zapata and Henao [4] statement, Santiago and Morales [5] promote a risk alpha to be included in the Essence kernel, since such a kernel only addresses risks when they occur. The inclusion of the risk alpha into the Essence kernel allows software engineering teams for monitoring and controlling risks before they occur.

In order to address Essence kernel problems related to the risk alpha exclusion, Zapata and Henao [4] and Santiago and Morales [5] have defined a proposal for the risk alpha. Zapata and Henao work consider some relationships between the risk alpha and the Essence kernel alphas. However, the introduction of the risk alpha into the Essence kernel requires additional relationship definitions among the Essence kernel elements—e.g., a new activity space providing support to perform activities for progressing the risk alpha and the completion criteria for such activity space. Consequently, in this chapter, we define a set of relationships between the risk alpha and the Essence kernel elements. Likewise, we explore additional relationships between the risk alpha and the Essence kernel alphas.

We expect new relationships defined in this Chapter allows for introducing the risk alpha into the Essence kernel. Therefore, teams can track and assess progress and health of software engineering endeavor risks. Semat Essence kernel allows software engineering teams for monitoring and controlling risks by tracking software engineering endeavor risks. Also, contingency actions for preventing potential risks can be defined and tracked [6].

This Chapter is organized as follows: in Section II we present the theoretical framework of the proposal; in Section III we present some background about the risk alpha and the problems surrounding its definition; in Section IV we propose a solution to the identified problems; finally, in Section V we discuss conclusions and future work.

## 2. Theoretical framework

Semat is intended to address the lack of a common ground of the software engineering by defining a theory composed of universal elements covering all software engineering endeavors [1]. Such a theory is composed of an Essence kernel and a language [2].



Essence kernel is categorized in three areas of concern: Customer, Solution, and Endeavor. Such areas represent a way to gather together the universal elements in the Essence kernel [2]. Such universal elements are defined as follows:

- Alphas represent “the things we always work with.” They are the most important dimensions we have to care about in a software engineering endeavor [2].
- Activity spaces represent “the things we always do” in a software engineering endeavor. They provide support to perform activities for progressing the Essence kernel alphas [2].
- Competencies are the skills required for performing software engineering activities [2].

Essence kernel alphas and their relationships are presented in Figure. 1. Also, Essence kernel activity spaces are presented in Figure 2.

Alphas are composed of states. Such states provide checklists with the criteria the team should fulfill for achieving them. Checklists provide guidance in the things to be done for progressing an alpha [2].

Activity spaces define completion criteria. They establish when an activity space is completed and they can be expressed in terms of alpha states [2].

We can review how a software engineering endeavor is advancing by recognizing the state an alpha has. Hence, alpha states lead to the Essence kernel actionability [1].

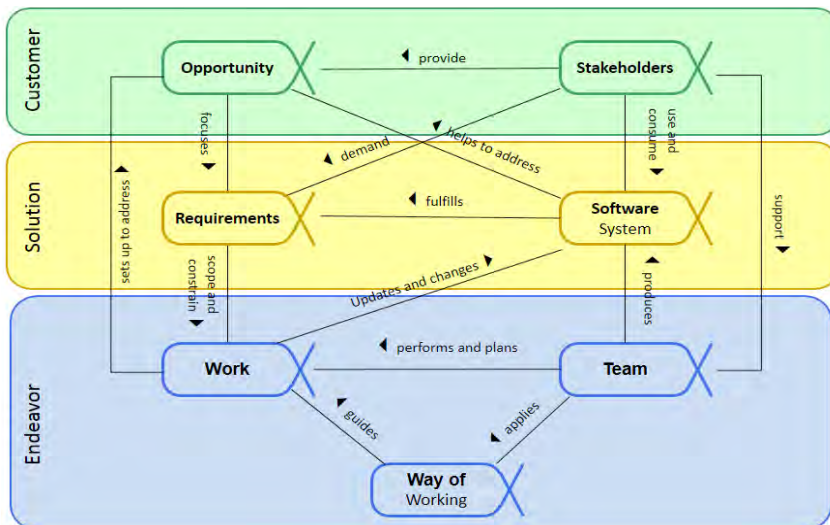


Figure 1. Essence kernel alphas. Source: [2].

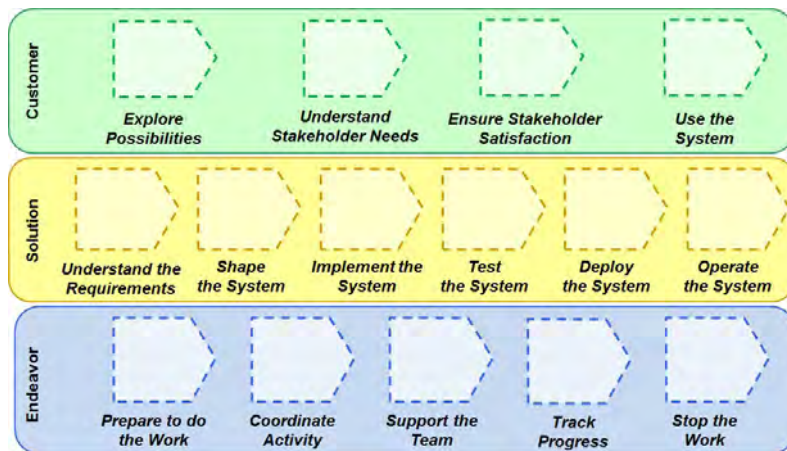


Figure 2. Essence kernel activity spaces. Source: [6].

### 3. Background

In this Section, we identify some problems related to the exclusion of the risk alpha from the Essence kernel. Also, we present Zapata and Henao [4] proposal for solving such problems.

According to Jacobson et al. [3] the risk alpha was neglected as an alpha of the Essence kernel, since they consider checklists in the Essence kernel already address risks threatening software engineering endeavors. They say “if you are having trouble achieving the Requirements Alpha Coherent state because there are conflicting requirements that you are having trouble solving, then this is an indication you may have a requirements risk” [3].

Zapata and Henao [4] recognize the fact Essence kernel helps to identify what dimension of a software engineering endeavor is suffering the impact of risks. However, they state the way risks are managed in the Essence kernel omit the Essence kernel actionable feature, since the possibility of tracking the progress of those risks in an explicit way is left unguided.

Jacobson et al. [3] statement supports Santiago and Morales [5] position about the Essence kernel when they say risks are only addressed when they occur. According to Keil et al. [7], such situation could lead development teams to failure when they are running a software engineering endeavor. They state the high failure rate of software engineering endeavors is related to the lack of risk management.

Zapata and Henao [4] identify some problems with alpha state checklist items where risks are addressed in the Essence Kernel. Such checklists are presented in Table 1.

**Table 1. Alpha state checklist items addressing risks [2,4].**

Alpha	State	Checklist item
Opportunity	Viable	"Risks are manageable"
Software System	Architecture selected	"Architecture selected that address key technical risks"
Work	Prepared	"Resource availability and risk exposure understood"
Work	Under control	"Work going well, risks being managed"

Some checklist items are supposed to address risks in the Essence kernel but they fail, since some of them use adjectives for qualifying risks—e.g., Opportunity viable state says "risks are manageable" [2]. Such qualification could lead to some misunderstanding, since the criteria for determining when a risk is manageable are excluded from the Essence kernel [4]. Zapata and Henao [4] state such qualification left unguided the things have to be done for achieving some alpha states, which runs counter to checklist definition.

In order to address problems described above Zapata and Henao [4] and Santiago and Morales [5] have defined a proposal for the risk alpha. Zapata and Henao [4] proposal is presented in Figure 3 and Figure 4.

Zapata and Henao [4] define five states of the risk alpha. The first state is "uncertain," for representing the starting point of a software engineering endeavor. At that moment, the risks threaten the software engineering endeavor but the team is unaware of the impact they will have and the likelihood of their occurrence. This situation should lead teams to understand the need for a management plan. So, they can progress in the risk alpha.

The second state of the risk alpha is "identified," allowing teams for identifying the risks threatening the software engineering endeavor and assessing whether is worthy or not the running of the software engineering endeavor [4]. Usually, at the beginning of a software engineering endeavor risks arise—e.g., unclear scope, unrealistic schedules and budgets, inadequate skills, and so on [8]—so they should be identified. By identifying those risks, the team should be able to establish a set of resources and work products to be used for performing the risk management. Identification process ends by making available a risk management plan [4].

The third state of the risk alpha is "understood," leading teams to perform both qualitative and quantitative analysis of the risks. By performing such analyses, the team can assess the impact of the risks and the likelihood of their occurrence [9]. The Output of

those processes allows stakeholders and team for understanding risk exposure and committing to address risk management throughout the software engineering endeavor [4].

Fourth state of the risk alpha is “planned,” allowing teams for planning actions in order to enhance opportunities and reduce threats, defining contingency actions by establishing how the team should act when a risk is materialized, defining actions by establishing how risks should be monitored and controlled, and establishing a budget for performing contingency actions [9].

Finally, the fifth state of the risk alpha is “under control,” for performing control and monitoring of risks—e.g., assessing whether contingency actions have reduced risks to acceptable levels—and waiting for new risks to arise [4, 9].

Alpha relationships proposed by Zapata and Henao [4] in Figure 4 ignore some relationships existing between the risk alpha and the Essence kernel alphas—e.g., risk alpha “understood” state is related to the commitment stakeholders and the team has for addressing risks. However, such relationship is excluded from the risk alpha relationships. Consequently, new risk alpha relationships need to be included for facilitating risk alpha inclusion in the Essence kernel.

Essence kernel activity spaces support activities for progressing in Essence kernel alphas. However, since the risk alpha is outside the Essence kernel, no activity spaces are defined for supporting activities that progress this alpha. Consequently, existing activity spaces need to be modified in order to include risk alpha and new activity spaces need to be defined.

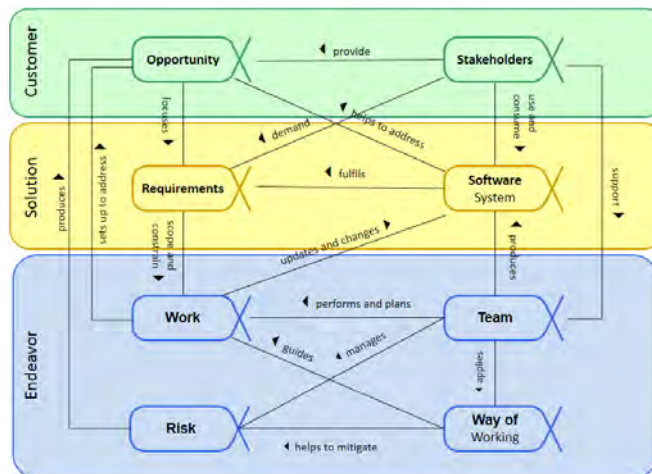


Figure 3. Zapata and Henao proposal for risk alpha relationships. Source: [4].

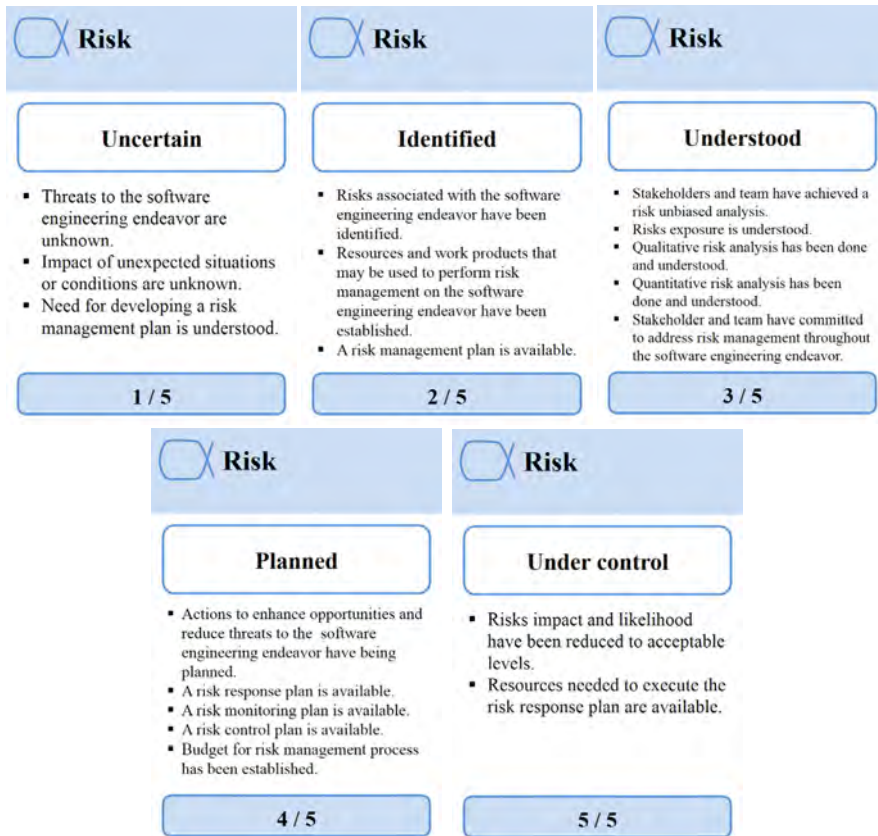


Figure 4. Zapata and Henao proposal for risk alpha. Source: [4].

## 4. Solution

In this Section, we define some relationships between risk alpha and other Essence kernel alphas. Likewise, we define some relationships between the risk alpha and the Essence kernel activity spaces. Such relationships lead us to define a new activity space providing support for performing activities that progress the risk alpha.

We define two new alpha relationships related to risk alpha and modify one relationship proposed by Zapata and Henao [4]. Such relationships are presented in Table 2.

In a software engineering endeavor, stakeholders and team commit to address risk. However, the role played by stakeholders and team is different. The team manage the risks could threaten the software engineering endeavor. Such management process in-

volves risk identification, management plan creation, quantitative and qualitative risk analysis execution, and so on. Stakeholders support the team throughout the management process. They provide the budget needed for performing risk management. Also, the decision making process about the risks of the software engineering endeavor is totally up to them.

We define the alpha relationship “Stakeholders commit to address Risk,” since Zapata and Henao proposal [4] consider the relationship between team and risk but excludes the relationship between stakeholders and risk. This relationship represents the decision making the process to continue with the software engineering endeavor once risks are identified. Such a process should be based on risk analysis performed by the team. Usually, this decision ends with the transition of risk alpha from the “identified” state to the “understood” state.

As we have mentioned before, the beginning of a software engineering endeavor is threatened by risks, but the team is unaware about the impact they will have and the likelihood of their occurrence [4]. Risks will remain in such state—“uncertain”—until some work is performed for identifying, managing, and controlling risks. So, teams should perform work to address software engineering endeavor risks. We represent this responsibility in the “helps to address” relationship between work and risk.

Semat defines an opportunity as the reason for running the software engineering endeavor. No software engineering endeavor is possible without an opportunity [2]. Accordingly, risks threatening the software engineering endeavor are threatening the opportunity too. So, the relationship between such alphas should be consistent. As a result, we consider the “produces” relationship between risk and opportunity proposed by Zapata and Henao [4] should be renamed as “threatens.” Alpha relationships described in Table II are graphically presented in Figure 5.

**Table 2. Risk alpha relationships**

Alpha	Alpha	Relation
Stakeholders	Risk	Commit to address
Work	Risk	Helps to address
Risk	Opportunity	Threatens

Despite some Essence kernel activity spaces provide support for activities progressing some risk alpha states, we lack an activity space for providing support to activities related to the software engineering risks identification, risk analysis, and risk understanding in the Essence kernel. Consequently, we define a new activity space for supplying such



a need. The activity space defined is presented in Figure 6. “Understand risks” activity space provides support to all of the activities performed by the teams in order to understand software engineering endeavor risk exposure, understand risks for achieving an unbiased risk analysis, and commit to address risks throughout of the software engineering endeavor.

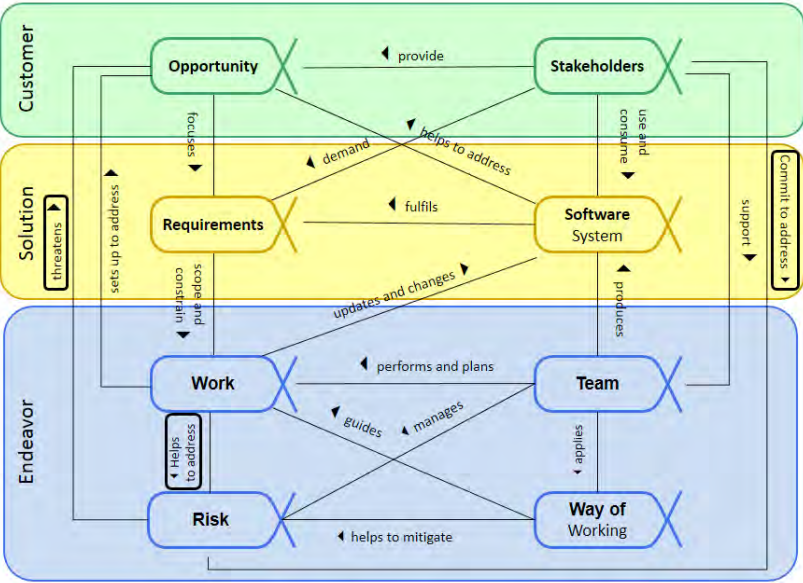


Figure 5. Zapata and Henao proposal for risk alpha relationships. Source: [4].

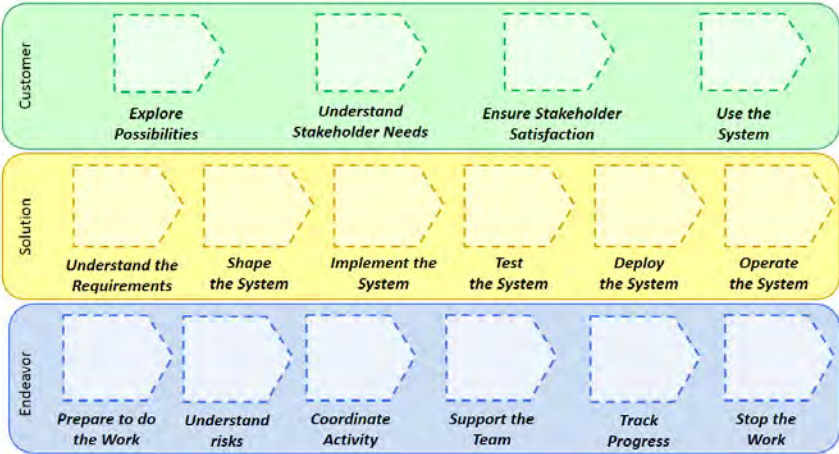


Figure 6. Understand risks activity space in Essence kernel. Adapted from: [4].

We modify the completion criteria of some Essence kernel activity spaces with clauses related to the risk alpha states for evidencing progress in the risk alpha. Such modifications are presented in Table 3. Also, completion criteria for the “understand risks” activity space is presented in Table 3.

**Table 3. Activity space modifications**

Activity space	Completion criteria	Added clause
Prepare to do the work	Team: Seeded, Way of Working: Foundation Established, Work: Prepared	Risk: Uncertain
Understand risks		Risk: Understood
Coordinate activity	Team: Formed, Work: Under Control	Risk: Planned
Track progress	Team: Performing, Way of Working: Working Well, Work: Concluded	Risk: Under control

## 5. Conclusions and Future Work

In this chapter, we exposed problems generated by the risk alpha exclusion from the Essence kernel and we propose some solutions to such problems.

First, we considered Zapata and Henao [4] proposal as starting point for the risk alpha. Then, we identified some ignored relationships between such proposal and other Essence kernel elements. Next, we defined three new alpha relationships, added new completion criterion clauses to some Essence kernel activity spaces and defined a new activity space. Finally, we allow teams for tracking and assessing the progress and health of software engineering endeavor risk by implementing such improvement to the Zapata and Henao [4] proposal. Hence, contingency actions for preventing potential risks can be defined and tracked [6].

As future work, we propose the exploration of other dimensions may affect the Semat Essence kernel actionable feature, since we only focused in the risk alpha. Also, Santiago and Morales [5] work should be explored and analyzed, since they pointed out several dimensions could affect the Essence kernel actionable feature. Finally, some relationships among such dimensions and other Essence kernel elements should be explored.

## 6. References

- [1] I. Jacobson, P. Ng, P. McMahon, I. Spence, S. Lidman, “The Essence of Software Engineering. Applying the SEMAT Kernel,” Indiana, United States of America: Addison-Wesley Professional, 2013.



- [2] Kernel and Language for Software Engineering Methods (Essence) version 1.1, OMG Std, 2015. <http://www.omg.org/spec/Essence/>.
- [3] I. Jacobson, P. McMahon, R. Racko, "24 Questions. Semat am Essence: They Why's, What's and How's to See the Difference," Ivar Jacobson International, November 2015. [Online]. <https://www.ivarjacobson.com/publications/white-papers/24-questions-semat-essence>.
- [4] C. Zapata, A. Henao, "Alfa Riesgo: Un elemento universal presente en todos los esfuerzos de ingeniería de software," Proc. Spanish edition. 4th International Conference on Software Engineering Research and Innovation (CONISOFT2016), Abril 2016, pp. 36-40.
- [5] B. Santiago, M. Morales, "ALPHAs basadas en el PMBOK: los elementos esenciales de un proyecto de software," Proc. Spanish edition. 4th International Conference on Software Engineering Research and Innovation (CONISOFT2016), Abril 2016, pp. 29-35.
- [6] A. Nehari, R. Mat-Zin, M. Houari, "Risk Management and Information Technology Projects," International Journal of Digital Information and Wireless Communications, vol 4, no 1, pp. 1-9, 2014.
- [7] M. Keil, P. Cule, K. Lyytinen, R. Schmidt, "A framework for identifying software project risks," Communications of the ACM, vol 4, Issue 11, pp. 76-83, 1998.
- [8] T. Addison, S. Vallabh, "Controlling Software Project Risks—an Empirical Study of Methods used by Experienced Project Managers," Proc. Annual Research conference of the South African institute of computer scientists and information technologists on Enablmenet through techonology (SAICSIT'02), South Africa, 2002, pp. 128-140.
- [9] PMI, A guide to the Project Management Body of Knowledge (PMBOK Guide), 2000 ed., Newtown Square, PA, United States of America: Project Management Institute, Inc., 2000.



# PART 3 TEACHING

---

# Chapter # 21

## Towards a Compilation of Problems in the Adoption of Agile-Scrum Methodologies: A Systematic Literature Review

**Janeth López-Martínez, Reyes Juárez-Ramírez, Carlos Huertas, Samantha Jiménez**  
School of Chemical Science and Engineering  
Autonomous University of Baja California  
Tijuana, Baja California, México  
{patricia.lopezm, reyesjua, chuertas,  
samantha.jimenez}@uabc.edu.mx

**Cesar Guerra-García**  
Polytechnic University of San Luis Potosi  
cesar.guerra@upslp.edu.mx

### 1. Introduction

Agile Software development is a new approach, which is gaining industry's attention in recent years. Most organizations are moving to the adoption of agile methodologies. Consequently, this tendency is due the continuous necessity of producing best solutions, fast development, and profitable software. Agile methods assume that changes in requirements are inevitable, and thus the software development cycle has to adapt to this fact. Moreover, software teams have to deliver value product to the customer as quickly as possible with fewer concerns on extensive planning and documentation [1].

Agile methodologies have been especially useful in projects with the especial characteristics [2]: small teams, short development calendars, constantly change in requirements, systems based on new technologies.

Successful agile adoption helps to produce higher quality software at a lower cost and enhances developer's moral than for example the traditional waterfall model approach.

Although most agile methodologies have been seen as a light process and easy to understand, but the adoption of this methodologies sometimes is very difficult. This is due that they are not obvious themselves, so it is hard to introduce agile methodologies in the culture of a company where people have a certain way of working during a long time.

Agile adoption always comes with special challenges and fundamental organizational changes that are necessary for successful outcome [1].

We found in literature many agile studies that were conducted to assess the merits and challenges of agile adoption. Some of the main issues reported are listed below [37, 38, 39, 40]:

- Team members reveal limited knowledge of agile method during the implementation phase.
- At the beginning of adoption, organization learning of the enterprise is not aligned with agile process adoption.
- At the beginning of adoption, requirements present different levels of abstraction, which derives in ambiguity dealing with conflicts in product quality.
- Developer fear caused by the transparency of skill deficiencies.
- The necessity for developers to be a “master of all trades” in order to get the desired product.
- Increased reliance on social skills.
- A lack of business knowledge among developers.
- The necessity to understand and learn values and principles of agile, not just the practices.
- Lack of developer’s motivation to use agile methods.
- Implications of devolved decision-making.
- The necessity for agile compliant performance evaluation.
- The lack of Agile-specific recruitment policies and suitably trained information technologies graduates.
- The necessity of integrating each pilot project with the project environment’s existing processes.
- The necessity of adding support for cross-team communication, especially in large teams that might be located in different geographical locations.
- Although Agile values code production is more than plan-driven processes, some developers tend to spend more time creating non-code artifacts and counting the number of meetings they attend than producing code.

- Developers, who view agile as micromanagement, perceive project management as being about due dates and missed deadlines.
- When an overzealous team moves quickly to Agile without careful planning, it usually results in a number of problems.
- Agile does not have separate coding and testing phases. Code written during iteration should be tested and debugged during the iteration.

Problems reported in each case study concern different aspects around the elements involved in the development process. We do not find a clear pattern of the problems presented in each case study. In order to integrate a guide for agile adoption, we propose a classification of those problems in the next four groups: people, process, project and company (organization).

In this chapter, we present a Systematic Literature Review (SLR) performed to identify a pattern of problems presented during agile methodologies adoption. In this way, we could correlate best practices extracted from another well-proved methodologies and process models such as CMMI, MoProSoft among others, to find solutions for agile problems.

This chapter is organized as follows. Section II gives a brief description of agile methodologies, emphasizing Scrum. Section III summarizes related work. Section IV describes the method used for the systematic literature review. Section V exposes the results of the SLR. In Section VI the problems in the Project category are described in detail. Section VII presents the conclusions and future work.

## 2. Fundamentals of Agile Methodologies

Software development methodologies have been the main focus of life cycle approaches to any project. Since 1940, there have been significant changes in software development paradigm, having approaches such as structured programming, object-oriented programming, and more recently extreme programming and aspect-oriented programming. Each evolutionary change introduces new ways of thinking and of analyzing problems, besides; it introduces strengths to the software development. In order to use these methodologies efficiently, it is important to follow defined process as they are formulated [3].

Agile methods have been developed as an effort to improve perceived and real weaknesses of the conventional Software Engineering. After decades of being utilized, agile

methodologies provide important benefits to projects. However, the methods are not applicable to all projects, products, people and situations [4]. In an industry context, introducing agile methodologies enable considerable changes in people's work habits; this is due these methodologies establish intentionally an opposite side with respect to the traditional software development approaches [5].

Many agile methodologies have been proposed, and they are still used in the software development industry; some examples are Extreme Programming, Adaptive Software Development (ASD), Scrum, Cristal, and others. However, our principal interest is in the Scrum adoption that is the most used nowadays.

## 2.1 SCRUM

According to recent studies [13], many companies have improved the quality of their products by applying Scrum practices; hence it is considered the most popular agile methodology. Scrum is basically an iterative, incremental and empirical process to manage and control the development of a project.

In Scrum, the customer is allowed to make changes to requirements at any time between sprints, this feature arises some challenges as issues cannot be predicted or is very difficult to tackle them in a planned way. Given this issue, Scrum methodology accepts that this problem cannot be completely solved or understood, therefore it focus on improving the team ability to deliver quick and useful responses [7].

Scrum is composed by three main roles: Scrum Team, Scrum Master and Product Owner. The Scrum Team, which must be self-organized and are responsible for the product development and the task time estimations for each team member for every sprint. Usually a sprint has a 2 to 4 week duration and it starts with a meeting so the product owner and team can understand what is going to be done in the following sprint [6] [8] [9]. The Scrum Master is the team mentor, and main responsible for the process, he/she is required to teach each member if required and help to solve issues that might arise during the project. The Product Owner represents the customer, and the generation of the backlog is the main activity, which basically represents a list of requirements that are sorted given their priority.

The duration of every Sprint is from 2 to 4 weeks and is the main activity of Scrum. Each Sprint starts with the planning meeting of the Sprint. The Product Owner and Teamwork get together to know what will do for the next Sprint [6] [8] [9]. The Figure 1 shows the main elements for the Scrum process.

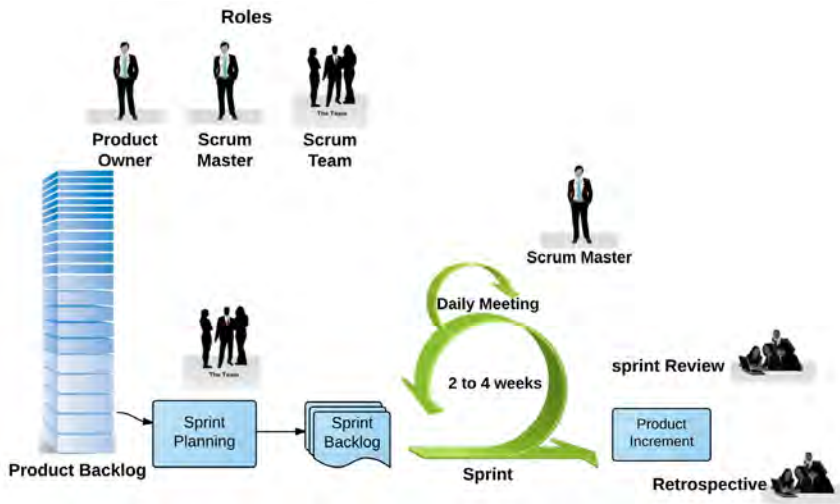


Figure 1. Scrum process, a light view.

Currently, it has been found that companies that implement Scrum can be benefited with increased profitability, communication, teamwork, among others. However, it is usually hard to adopt an agile methodology because it requires lots of commitment from the team and this is not always easy. For this reason, we carried out a systematic literature review to identify which are the most common problems that arise during Scrum adoption. The methodology used for this review is detailed below.

### 3. Related Work

In the software engineering research literature, a number of researchers have worked on studying the adoption of agile methodologies and their implementation, such as M. Dalhem et al. [11], I. Inayat et al. [58], A. Kanane [59], E. Cardoso et al. [60], M. Kaisti [61] and T. Dingsøyr et al. [62].

I. Inayat et al. [58] present a systematic literature review on practices and challenges of Agile Requirements Engineering (ARE). Their purpose is to learn how traditional Requirements Engineering issues are resolved using agile methods. There 17 ARE practices are described that they found to be adopted in agile software development. For each practice, identify its potential respective challenges, also found that while Agile Requirements Engineering practices help counter the challenges experienced in traditional Requirements Engineering. On the other hand, they also introduce several limitations for achieving an adequate balance between agility and stability and ensuring sufficient competence of cross-functional development teams.

The challenges that ARE imposes to project organization include minimal documentation, budget and schedule estimation, inappropriate architecture, neglect of non-functional requirements, waste management, customer unavailability and contractual issues.

E. Cardoso et al. in [60] describe a systematic literature review, which aims to find scientific evidence of the correlation between the use of Scrum and productivity in software projects. Moreover, they found other aspects that could be affected by the implementation of Scrum, such as product quality, client satisfaction, cost reduction and team motivation, which are not necessarily related to productivity but represent general success characteristics in software development projects.

A. Kanane [59] performed an investigation about the challenges related to the adoption of Scrum. This study was conducted over the case of a company operating in the financial IT sector and that has been using Scrum for 5 years. The survey focuses on the challenges related to the transition governance while adopting Scrum, investigating the challenges connected to the execution of the process, the work routines, and tools and techniques used to develop IT based projects with the Scrum method. The results were structured around three categories of challenges: challenges in relation with the prerequisites for executing the work, with the sprint management, and with the team dynamic.

The main result of this study shows that adopting Scrum is an ongoing process that never ends. Moreover, there are no standards or guidelines discussing how to adopt Scrum successfully.

M. Dalhem et al. [11] conducted a mapping study trying to understand which agile practices are the most used in the industry under different circumstances, such as different project types, domains, or processes. The results of this study show that there are practices that are used more often and that the domain and the process influence the application of different practices.

The following is a list of universal agile practices:

1. Quality check
2. Refactoring
3. Customer involvement
4. Unattached communicative teams



5. Validation practice
6. Learning loop
7. Outcome review
8. Planning meeting
9. Time boxing
10. Common knowledge
11. Progress monitoring
12. Product vision
13. Evolving and hierarchical specification
14. Continuous integration/deployment
15. Delivering frequent releases
16. Small cross-functional teams
17. Daily discussion
18. Continuous specification analysis

T. Dingsøy et al. in [62] examined publications and citations to illustrate how the research on agile has progressed in the 10 years following the articulation of the agile manifesto. They summarized prior research and introduce contributions on agile software development. The number of special issues devoted to agile development is also an indication of the intense interest displayed in software engineering and other related fields, notably information systems.

M. Kaisti in [61] conducted a literature review and a mapping study to bring forth what is known about agile methods in embedded systems development and to find out if agile practices are suitable in this domain and what evidence is there to support the findings. There was found that there are embedded domain-specific problems about agile methods that need to be solved before agile methods can be successfully applied to the embedded domain.

As we can see, some systematic literature reviews have been done resulting in the identification of specific problems, but not necessarily advising a classification of them. Our present work aims to give a classification of the main problems presented when adopting or implementing agile methodologies, especially Scrum.

## 4. Methodology for slr

Our SLR was guided by Kitchenham methodology [10]. The next sub-sections describe the method.

### 4.1 Formulating research questions

This study had the objective of identified problems in adopting agile methodologies such as Scrum. To conduct this study, we establish the following research questions (RQ):

RQ1: What are the main problems for adopting agile methodologies?

RQ2: What are the main problems for adopting Scrum?

The first question emphasize on identify problems for adopting agile methodologies in general. The second question focuses on Scrum methodology, particularly on problems for adopting this process.

### 4.2 Search strategy

We considered the following scientific electronic libraries: IEEE Xplore, Science Direct, ACM DL, and Springer Link.

We search publications from 2012 to 2016, trying for specialized journals and conferences.

We used a search string composed by keywords and logic connectors, dealing with the following search string:

(Agile Methodology OR Scrum) and (Adoption Problems OR Adoption Issues OR Adoption Challenges)

We searched in all electronic data bases the mentioned search string in three parts of a document: keywords, abstract, full text (body of the paper).

We used the following search strategy to recognize the most significant papers for this SLR:

- Selecting the keyword for research
- Looking in the digital library, trying with research keywords based on inclusion and exclusion criteria.

- Examining each paper through title and abstract.
- Downloading papers which cover search criteria.
- Screening each paper, taking a fast view of the sections in a paper.

### 4.3 Defining inclusion and exclusion criteria

As we mentioned earlier we selected papers must be published between January 2012 and July 2016. We took into account English and Spanish papers. In some cases, we found short and full versions of papers, but only full versions were reviewed. We included papers with a practical approach, in other words, we excluded theoretical papers.

## 5. Results from the review

In Table 1 we are showing the activities realized for our search. We searched for keywords “Adoption, Adoption Problems, Scrum, Agile Methodologies, and Challenges”, Using those keywords we found 281 papers related to our research in the most popular scientific research libraries. These papers were reviewed at the first step to the level of “Title”, reducing the amount to 119. Second step was to review the “Abstract”, reducing the amount to 93. Finally, taking a look at the general view of the full papers, reducing them to 36, which are the most relevant for our research.

**Table 1. Selected papers**

Database	Original search	Title	Abstract	Fast view	Selected
IEEE Xplore	39	33	27	20	18
Science Direct	79	42	28	14	7
ACM DL	111	21	18	5	5
Springer Link	52	23	20	17	6
Total	281	119	93	56	36

In Table 2 we are showing the 36 papers selected for our research. The table details the data of each paper as ID of the paper, the name of paper, author’s names, the electronic library where it was obtained and publication’s year.

**Table 2. SELECTED PAPERS**

Title	Author	Database	Year
Agile Practice in Practice: A Mapping Study [11]	M. Dahlem	ACM	2014
Agile Methods, Organizational Culture and Agility: Some Insights [12]	Lakshminarayana Kompeya	ACM	2014

This table continues on the following page————>

Title	Author	Database	Year
How to Make Agile UX Work More Efficient: Management and Sale Perspectives [13]	Kati Kuusinen / Kaisa VaananenVainio Mattila	ACM	2012
Adopting Agile Software Development: Issues and Challenges [1]	Hassan Hajjdiab and Al Shaima Taleb	ACM	2011
Agile Beyond Software Development [14]	Dan X. Houston	ACM	2014
A multi-faceted Roadmap of Requirements Traceability Types Adoption in SCRUM: An Empirical Study [15]	Ghada Alaa / Zeinab Samir	IEEE	2014
Influences on Agile Practice Tailoring in Enterprise Software Development [16]	Julian M. Bass	IEEE	2012
Scrum Anti-patterns – An Empirical Study [17]	Veli-Pekka Eloranta, et al.	IEEE	2013
ScrumBut, but Does It Matter? [18]	Ville T. Heikkilä, Maria Paasivaara and Casper Lassenius	IEEE	2013
Where Is Scrum in the Current Agile World? [19]	Georgia M. Kapitsaki and Marios Christou	IEEE	2014
An Empirical Study of Social Success Factors for Agile Software Development [20]	Evelyn van Kelle, Per van der Wijst. Aske Plaat. JoostVisser	IEEE	2015
Agile Adoption Story from NHN [21]	Eunha Kim and Seokmoon Ryoo	IEEE	2012
Beyond Mainstream Adoption: From Agile Software Development to Agile Organizational Change [22]	David Bustard	IEEE	2012
How We Successfully Adapted Agile for a Research Heavy Engineering Software Team [23]	Alfred A. Lobber , Kyran D. Mish	IEEE	2013
The Maturation of Agile Software Development Principles and Practice: Observations on Successive Industrial Studies in 2010 and 2012 [24]	David Bustard, George Wilkie, Des Greer.	Science Direct	2013
Operational release planning in large-scale scrum with multiple stakeholders – A longitudinal case study at F-Secure corporation [18]	Ville T. Heikkilä, Maria Paasivaara, et al	Science Direct	2014
Obstacles to decision making in Agile software development teams [25]	Meghann Drurya, Kieran Conboyb, Ken Power	Science Direct	2012
Agile Principles and Achievement of Success in Software Development: A Quantitative Study in Brazilian Organizations [3]	Paulo de Souza B., André Luiz Zambaldea et al	Science Direct	2014
The impact of inadequate and dysfunctional training on Agile transformation process: A Grounded Theory study [26]	Taghi Javdani Gandomania, et al	Science Direct	2014
Towards optimal software engineering: Learning from agile practice [24]	David Bustard George Wilkie DesGreer	Springer Link	2013
The evolution of agile software development in Brazil [27]	Claudia de O. Melo et al	Springer Link	2013
The Role of Communication in Agile Systems Development: An Analysis of the State of the Art [28]	Markus Hummel, Dr. Christoph Rosenkranz. Dr. Roland Holten	Springer Link	2012

This table continues on the following page————>

Title	Author	Database	Year
Scrum adoption and architectural extensions in developing new service applications of large financial IT systems [29]	Toumas Ihme	Springer Link	2013
Evaluating the impact of an agile transformation: A longitudinal case study in a distributed context [30]	Kirsi Korhonen	Springer Link	2013
Strengths and barriers behind the successful agile deployment-insights from the three software intensive companies in Finland [31]	Minna Pikkarainen Outi Salo Raija Kuusela Pekka Abrahamsson	Springer Link	2012
Necessary Skills and Attitudes for Development Team Members in Scrum [32]	Penprapa Bootla et al,	IEEE	2015
Obstacles to efficient daily meetings in agile development projects: A case study [33]	Viktoria Gulliksen Stray et al.	IEEE	2013
Cost and effort estimation in agile software development. Optimization, Reliability, and Information Technology [43].	R Popli and N Chauhan	IEEE	2014
A sprint-point based estimation technique in scrum [42].	R Popli and N Chauhan	IEEE	2013
Could social factors influence the effort software estimation [44].	V. Lenaarduzzi	IEEE	2015
Adjusting Story Points Calculation in Scrum Effort & Time Estimation [45].	H. Zahraoui, M. Abdou, and J. Idrissi,	IEEE	2015
Identification of Inaccurate Effort Estimates in Agile Software Development [47]	Florian Raith, Ingo Richter, Robert Lindermeier, and Gudrun Klinker	IEEE	2013
On using planning poker for estimating user stories [48].	Viljan Mahnic and Tomaz Hovelja	Science Direct	2012
The Impact of Scrum on Customer Satisfaction: An Empirical Study [50].	B. Cartaxo, A. Araujo, A. Barreto, and S. Soares,	IEEE	2013
Comparison of Functional Size Based Estimation and Story Points, Based on Effort Estimation Effectiveness in SCRUM Projects [51].	E. Ungan, N. Cizmeli, and O. Demirors	IEEE	2014
Estimating, planning and managing Agile Web development projects under a value-based perspective [53].	C. J. Torrecilla-Salinas, J. Sedeño, M. J. Escalona, and M. Mejias	Science Direct	2015

The results of the SLR were organized into the categories proposed by Shahane et al, taking as reference the factors model of agile methodologies adoption proposed by in [34]: people, processes, project, and organization aspects in the company. This model was adapted by M. Shuhuay et al. in [35] categorizing the factors of the agile method adoption, which are shown in Figura 2.



Figure 2. Factors for Agile Methods' adoption

Table 3 shows the problems found in the SLR, focusing on the adoption of agile methodologies and Scrum.

Table 3. Problems in the Agile Migration

Key issues in the agile migration	Category	Papers
Organizational culture does not support agile ways of working	Organization	[17], [19],[27]
Lacks of capacity to change the organizational culture	Organization	[12], [17]
Organizational problems	Organization	[17], [19]
Lack of management support	Organization	[3]
External pressure to use traditional practices	Organization	[17], [27]
Lack of collaboration and communication with the customer	People	[19], [26]
Lack of training of the Product Owner and the customer	People	[27]
Team size	People	[12], [19], [24], [29]
Team unaligned	People	[20]
Equipment capacity	People	[3]
Rotating team members	People	[27], [29]
Lack of experience with agile methods	People	[17], [27]
Availability of trained personnel	People	[20], [32]
Lack of effective communication	People	[20], [26],[27]
Lack of understanding of agile values	People	[26]
Inadequate and dysfunctional training	People	[17]
General resistance to change	People	[20], [27]
Lack of commitment to decisions	People	[32]
Continued involvement with the client	People	[20]
Project size	Project	[12], [19], 29]
Agility degree	Process	[20]
Anti-patterns	Process	[17]

Figure 3 summarizes the classification of the problems found in SLR. Considering our findings, we can state that the literature reports more people problems.

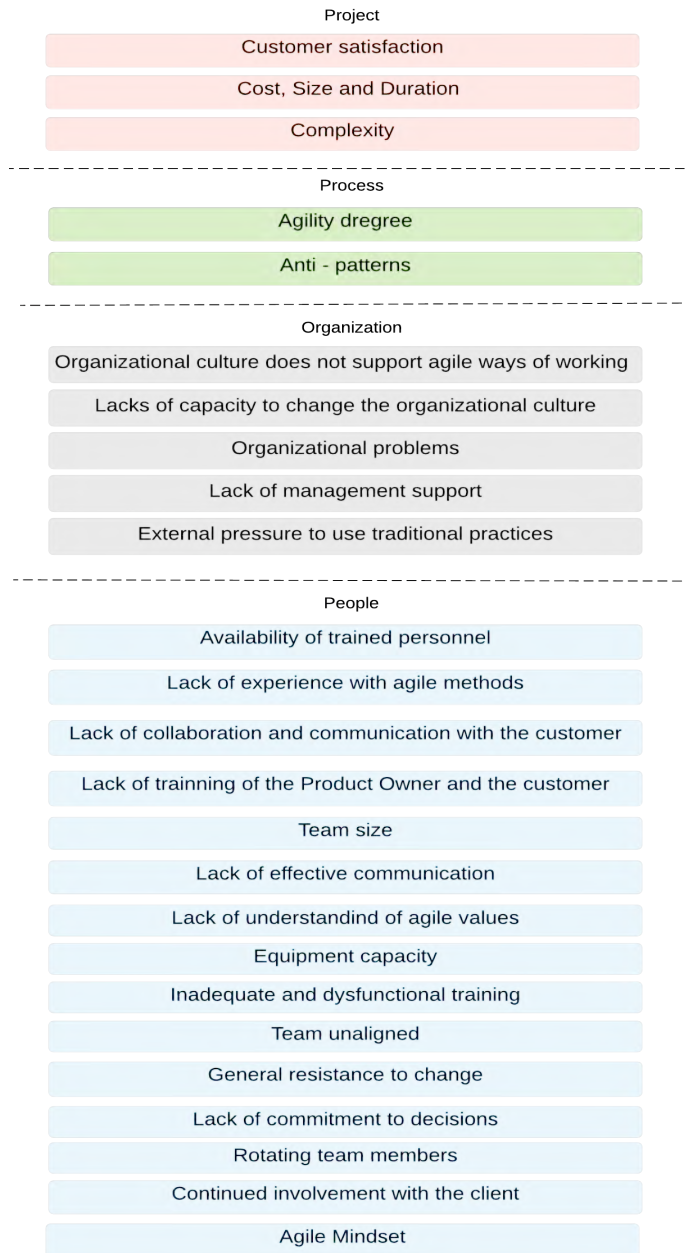


Figure 3. SLR problems classification.

In the following sections we analyze each of the problem categories.

## 5.1 Organizational aspects

Organizational culture is a shared belief system that permeates an organization or subunit and ultimately influences the actions of people and workgroups [50].

The successful adoption of agile methodologies involves many factors; organizational culture is one of them, and this factor has great importance in the adoption of Scrum. We reviewed which the problems are presented in this category, we found the followings: the organizational culture does not support forms of agile work [17, 19, 27, 3]; it lacks the capacity to change the organizational culture [17, 3, 26, 36]; organizational problems [24, 20, 3, 26]; lack of support from the heads of companies [17, 19]; and external pressure to use traditional practices [17, 27].

The Agile methodologies must be used within an agile culture that is characterized by a broad support for the negotiations, a capacity for change, the collaboration and the continuous exchange of experiences and knowledge.

According to W. Schneider [55], there are four kinds of organizational culture: collaboration, Control, Cultivation, Competence. It is clear that certain types of culture would be detrimental or make it impossible to successfully use an agile method [50], due that it is important to identify if the organization supports the change toward an agile culture. For transforming from traditional to agile methods, management style should be changed from “command and control” to “leadership and collaboration” [25].

## 5.2 People

Achieving a cooperative process based on the communication and collaboration between members, who value and trust each other, is critical for the success of agile methods. Human aspects most of the time act as an obstacle in agile adoption [25].

The change from a traditional development process like waterfall to an agile method needs a big change to people thinking and of their behavior, for this reason people are a critical factor in the adoption of Scrum [32, 26]. This is due to the fact that this methodology believes in self-organization of the members of the development team. The majority of projects fail because the team lack of effective communication, and the communication style is more important than the frequency of the communication, we can see that the informal communication can improve the success of the project.



There are multiple studies about the role of the people in the agile adoption. Problems found are as follows: Lack of cooperation and constant communication with the client [27, 29, 19]; lack of training of the Product Owner and the client [27]; size of the team [19, 29, 12]; teams not aligned [20]; the abilities of the teams [3]; rotation of members of the team [29]; lack of experience with the agile methods [17, 27]; availability of trained personnel [17, 19]; lack of effective communication and misunderstandings [20, 32]; lack of comprehension of the Agile values [20, 32, 26]; inadequate and dysfunctional training [26]; overall resistance to change [17]; lack of commitment to the decisions [20, 27]; lack of agile mindset [46] and lack of constant participation with the customer [32].

The lack of necessary skills, in fact, it can affect the success of the Scrum team. Bootla et al. [32] propose what these skills should be for each team member; such skills and attitudes can be used to continuously improve the team capabilities. These skills are essential to help teams to achieve their goals and success.

Their proposed skills and attitudes are categorized into 3 types as follows:

- (1) **Technical Skills:** Technical skills are directly related to software development activities.
- (2) **Soft Skills or People Skills:** Since Scrum focuses on a self-organizing team, the proposed soft skills are derived from soft skills related to working with others, to manage the project, and soft skills related to individuals.
- (3) **Attitudes:** The attitude affects how people behave and perform tasks. Scrum is unlike other traditional software development methodologies, and each team member attitudes toward software development may impact his/her performance.

## 5.3 Project

This category comprises the customer satisfaction, cost, duration, size, and complexity [35]. Projects that are over-budget, delivered late, and fall short of user's expectations have been a common problem area related to the accuracy of estimation for years, estimating is a serious challenge and estimates are frequently wrong [45] [46]. In the first version of this survey [56], few studies were found in the literature in this category since the majority of the papers found are focused on organizational aspects and people, so that we focusing in this category. This category is described in detail in section VI.

## 5.4 Process

Changing attitudes and moving to agile activities from rigid, adequate and planned activities is not available without spending enough time, effort and investment [25].

In a study presented by Lober and Mish [23], some problems that appear in the early stages of the Scrum adoption are identified, such as following: lack of delivery of user stories, lack of confidence, as well as the times in the planning meetings, daily meetings and retrospectives can be too long with little value to the attendees.

Some specific situations are:

- The larger the team the larger meetings are and it is harder to provide value to all participants.
- A medium speed and the points for user stories are insufficient to plan a sprint because of the size of the big team, the fluctuating participation and the specialization of people.
- Writing concrete user stories can be difficult due to the inherent uncertainty.
- Moving the adoption forward to an adequate pace requires transparency, inspection and adaptation.

All the team involved in the Scrum process must understand what is expected of everyone. It is very effective to explain why these expectations are set in terms of other people who are relying on the information.

V. Eloranta, K. Koskimies, T. Mikkonen, and J. Vuorinen [17] refer to some specific problems in adopting Scrum: (1) The harmful deviations from recommended Scrum practices and; (2) recommended Scrum practices that are for some reason unsuitable in a particular context. V. Eloranta, K. Koskimies, T. Mikkonen, and J. Vuorinen made some recommendations for Scrum's adoption:

- For the first type of problem, a company starting in Scrum should be aware of common deviations that may seem reasonable, but which are actually harmful.
- For the second type, the understanding of deviations from the norm, Scrum books provide information for improving the methodology to suit the purposes of the companies that develop software.

Both types of deviations named Anti-patterns are harmful to the projects.

## 6. Problems in the Project category

### 6.1 Customer Satisfaction

The selection of a software development methodology is a vital activity in any software project. It has a great impact on customer satisfaction and business welfare [52].

Customers play a critical role in the success of agile methods and they should be responsive, collaborative, authorized, committed and knowledgeable, having such customers is not easy and this role could be a barrier in the success of agile projects especially when they join the team for the first time. [25].

Customers probably have different definitions of “success” within a software project. B. Cartaxo et al in [50] realized a survey that aimed to determine whether there is any impact on customer satisfaction caused by the Scrum adoption. Seven critical factors are considered for customer satisfaction, and consequently, for project success: time, goals, quality, communication and transparency, agility, innovation and benchmark.

Their results indicate that it was not possible to establish any evidence that using Scrum may help to achieve customer satisfaction and, consequently, to increase the success rates in software projects.

Another study realized by M. Kohlbacher [57] highlights the negative impact of the change in requirements on customer satisfaction. Requirement changes during ongoing product development projects threaten the desired outcome of the project and therefore also customer satisfaction.

### 6.2 Cost, Size and Duration

Starting a professional software development project soon raises some critical questions such as: How much will the project cost? When will it finish? How much effort must be invested in it? Will the investment be returned soon? What are the features our customers really need? Being able to answer these questions and some others related to them is crucial for designing business strategies [53].

Estimation of cost, effort, size and duration of a software project is a difficult task. Accurate estimations of software are critical for both developer and customer [19, 43].

It has been observed that the current estimation method in Scrum mostly relies on historical data from past projects and expert opinion but in the absence of historical data and experts, these methods are not efficient. Ignorance of estimation methods may cause serious problems like exceeding the budget, not delivered on time, poor quality and not right product [42].

According to R. Popli [42], several problems are in existing estimation and tracking methods for Scrum software developments. Some of them are cited next:

- The first problem is effort estimation: Effort and time estimation at the initial stage of an agile project is a difficult and challenging task due to volatility and changing in customer requirements [45]. Effort estimation in Scrum is mainly based on story points which are subjective measures and mostly lead to inaccurate estimates [45]. Accurate estimations are essential for successful management, most of the unsuccessful software projects fail because of inaccurate estimations [51].
- The second problem is about Release Date Estimation: the Release planning is the activity to calculate the actual release date so that the final product is handed over into use for the customer. In Scrum Estimation technique, a release plan is made but it doesn't consider various factors like velocity, the cost-benefit ratio [45].
- The third problem is that story points cannot be easily related to the time duration because it represents the amount of work and the velocity differs from team to team.
- The fourth problem is that because a story point is a relative value, the total story point value can fluctuate with a slight variation in the baseline story point. To set the base use story, the agile team finds the simplest user story and determines story points of other user stories based on the baseline. If the baseline story point changes, other story points also have to be changed [42].

Scrum Estimation methods may lead to the errors in case of the inexperienced agile team.

R. Popli and N. Chauhan [42] propose various project and people-related factors that affect the cost, effort and duration of a software project, this factors are showing in Figure 4.

PROJECT AND PEOPLE RELATED FACTORS	
Project Related Factors	People Related Factors
<ul style="list-style-type: none"> <li>• Quality Requirement</li> <li>• Type of Project</li> <li>• Hardware and Software Requirement</li> <li>• Easy of Operation</li> <li>• Data Transaction</li> <li>• Multiple Site</li> </ul>	<ul style="list-style-type: none"> <li>• Communication Skill</li> <li>• Familiarity in Team</li> <li>• Managerial Skill</li> <li>• Security</li> <li>• Working Time</li> <li>• Experience of Previous Project</li> <li>• Technical ability</li> </ul>

Figure 4. R. Popli and N. Chauhan's factors

H. Zahraoui, M. Abdou, and J. Idrissi in [45] propose to adjust the story points using three adjustment factors; these factors are: Priority Factor, Story Size Factor and Complexity Factor.

**Priority Factor (PF):** User stories (US) are sorted in the product backlog (PB) from high priority to low priority. According to their survey prioritization of features in release, planning is based on two dimensions that are *urgency* and *business value*.

The **urgency** of a user story depends on its delivery date imposed by the product owner. Therefore, most urgent user stories must be integrated into the first sprint.

**Business Value** means the amount of revenue that might be generated or lost by a user story. So in Scrum user stories that have the highest Business Value must be delivered in the first sprint.

**Story Size Factor (SF):** The story size is a factor that affects the accuracy of scrum estimations because very large stories are more difficult to estimate than small ones. One of the best practices adopted by scrum team is to split very large stories into small ones but sometimes they don't have enough details about these stories

**Complexity Factor (CF):** Complexity introduces uncertainty to the estimation of each story in the Product Backlog, so more complexity means more uncertainty [45].

A lot of information should be taken into account to estimate the effort, such as the project size, the domain, and many other factors that may significantly influence the estimation [44].

### 6.3 Complexity

This section and section B are related closely, complexity refers that how complex is to develop a project. Technical complexity includes a number of aspects such as numbers of technologies are involved, the number of the technical interfaces. Management complexity includes project staffing and management etc. Complexity is the major aspect in the estimation of a project, the complexity of the project increase *cost*, *size* and *duration* [42].

Scrum does not provide a unique estimation technique. However, the most used technique is Planning Poker [47] [48]. Planning Poker is a group-based estimation technique for estimating the size of user stories and developing release and iteration plans [45], [46, 48].

The game utilizes playing cards, printed with numbers based on a modified Fibonacci sequence similar to 1-2-3-5-8-13-21- 40-100. The Product Owner and all members of the development unit meet to discuss the Product Backlog requirements for the purpose of reaching consensus-based estimations [45, 48, 49].

Planning poker brings together multiple expert opinions to do the estimation. Because these experts form a cross- functional team from all disciplines on a software project, they are better suited to do the estimation task than anyone else. Group discussion is the basis of planning poker, and those discussions lead to an averaging of sorts of the individual estimates [54].

Planning poker has many benefits. However, this method is not efficient because the result is always based on the observation of an expert and his/her experience. The story-points is a relative value and cannot be easily related to the time duration [43], [47], [48]. Moreover, the team member decision is unclear because they take into account only the complexity in general.

Usually in the practice, every member of the team assigns a complexity of a number of points (in this case five points) to a task but is unclear how the team member assigns the punctuation. Figure 5 shows this scenario. It is necessary to break down this variable (complexity) into its elements, in order to establish clearly how he/she chose the decision. By the moment, the evaluation is based on factors that are difficult to identify.

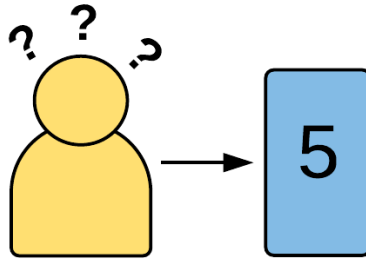


Figure 5. Planning Poker decision

Taking into account these circumstances, we can deduce that there is a strong need to analyze the different factors that affect the estimation of the Scrum project.

Moreover, there are a lot of factors that can cause uncertainty in effort estimation using story points. Those factors are usually related to the nature of the project to be developed, the context, and the knowledge of the domain and last, but not least, social factors such as the interaction in the development team also should be taken into account [44].

A set of social factors has been identified by V. Lenarduzzi [44]: Language and cultural differences, communication, team structure, communication process, work pressure, team size, competence level, familiarity in the team, managerial skill, working time, experience of previous projects, and technical ability.

## 7. Conclusions and future work

In this chapter, we presented a Systematic Literature Review about agile methodologies adoption, with a principal focus on Scrum. We emphasize different problems presented in agile adoption, especially on Scrum adoption.

We found that agile adoption always comes with special challenges; consequently, changes in the organization are critical to a successful outcome. The organization's culture is one of the main causes of resistance to change likewise other factors in making agile methodologies such as people, which is the largest existing impediment to the adoption. These problems must be treated to improve the adoption of agile methodologies and changing traditional development process to agile methods.

Taking into account our experience working with traditional life cycle models and process models, we find some best practices that can be a complement to Scrum. In previous publications [41] we presented a first proposal to adapt documenting practices

into the agile process for mobile applications, specifically in the requirements gathering and analysis and design stages.

In the project category, we can see that planning of the project is a crucial activity for the successful of the project. In the other hand, the estimations are important, as they are the basis for planning the next release in terms of prioritizing features and staffing the development. As we can see, a big set of information should be taken into account to estimate the effort and many factors that may significantly influence the estimation. These circumstances allow us to see that there is strong need to analyze the factors that affect the estimation of the Scrum project team for that we are focusing in this area.

As future work, we are preparing a model for tasks estimation in Scrum by considering the complexity in the main important factors that compose it.

Furthermore, we are preparing a framework in which we will establish our own values and practices for Scrum and some complementary practices not explicit in the methodology but considered as necessary to make an easy adoption. We are formulating a model based on Bayesian Networks to represent all the complexity's factors and their relationships. This model will allow us to assist novice software developers and small companies in estimating tasks.

## 8. References

- [1] H. Hajjdiab and Al Shaima Taleb, "Adopting Agile Software Development: Issues and Challenges," *International Journal of Managing Value and Supply Chains*, vol. 2, no. 3, pp. 1–10, 2011.
- [2] P. Letelier, "Metodologías Ágiles en el Desarrollo de Software," in *de Valencia, Valencia*, 2009.
- [3] P. H. D. S. Bermejo, A. L. Zambalde, A. O. Tonelli, S. A. Souza, L. A. Zuppo, and P. L. Rosa, "Agile Principles and Achievement of Success in Software Development: A Quantitative Study in Brazilian Organizations," *Procedia Technology*, vol. 16, pp. 718–727, 2014.
- [4] R. Pressman, *Ingenieria de Software Un Enfoque Práctico*. Mc Graw Hill, 6ta ed., 2008.
- [5] S. Overhage and S. Schlauderer, "Investigating the Long-Term Acceptance of Agile Methodologies: An Empirical Study of Developer Perceptions in Scrum Projects," in *2012 45th Hawaii International Conference on System Sciences*, pp. 5452–5461, IEEE, jan 2012.
- [6] M. Davila Muñoz, *Desarrollo de una especialización de moprosoft basada en el método ágil Scrum*. PhD thesis, Universidad Nacional Autonoma de México, 2008.



- [7] D. Pauly and D. Basten, "Do Daily Scrums Have to Take Place Each Day? A Case Study of Customized Scrum Principles at an E-Commerce Company," *Hawaii International Conference on System Sciences*, pp. 5074–5083, 2015.
- [8] K. Schwaber and J. Sutherland, "The scrum guide," *Scrum. Org*, October, vol. 2, no. July, p. 17, 2011.
- [9] M. Gannon, "An Agile Implementation of SCRUM," *Proceedings of the IEEE Aerospace Conference, Big Sky (MT), USA*, 2-9 March, 2013, pp. 1–7, 2013.
- [10] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in J. Clerk Maxwell, *A Treatise on Electricity and Magnetism*, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
- [11] M. Dahlem, "Agile Practices in Practice - A Mapping Study," 2014.
- [12] L. Kompella, "Agile methods, organizational culture and agility: some insights," *Proceedings of the 7th International Workshop on Cooperative and Human Aspects of Software Engineering -CHASE 2014*, pp. 40–47, 2014.
- [13] K. Kuusinen and K. Väänänen-Vainio-Mattila, "How to make agile UX work more efficient: management and sales perspectives," *Proceeding NordiCHI '12 Proceedings of the 7th Nordic Conference on Human-Computer Interaction: Making Sense Through Design*, pp. 139–148, 2012.
- [14] D. X. Houston, "Agility beyond Software Development," pp. 65–69, 2014.
- [15] G. Alaa, "A multi-faceted Roadmap of Requirements Traceability Types Adoption in SCRUM: An Empirical Study," pp. 1–9, 2014.
- [16] J. M. Bass, "Influences on agile practice tailoring in enterprise software development," *Proceedings - Agile India 2012, Agile India 2012*, pp. 1–9, 2012.
- [17] V. Eloranta, K. Koskimies, T. Mikkonen, and J. Vuorinen, "Scrum Anti-Patterns – An Empirical Study," *2013 20th Asia- Pacific Software Engineering Conference (APSEC)*, vol. 1, pp. 503–510, 2013.
- [18] V. T. Heikkilä, M. Paasivaara, K. Rautiainen, C. Lassenius, T. Toivola, and J. Järvinen, "Operational release planning in large-scale scrum with multiple stakeholders — A longitudinal case study at F-Secure corporation," *Information and Software Technology*, vol. 57, pp. 116–140, 2014.
- [19] M. Kapitsaki and M. Christou, "Where Is Scrum in the Current Agile World?" *Proceedings of the 9th International Conference on Evaluation of Novel Approaches to Software Engineering*, pp. 101–108, 2014.
- [20] E. V. Kelle, J. Visser, A. Plaat, and P. V. D. Wijst, "An Empirical Study into Social Success Factors for Agile Software Development," *2015 IEEE/ACM 8th International Workshop on Cooperative and Human Aspects of Software Engineering*, pp. 77–80, 2015.
- [21] E. Kim and S. Ryoo, "Agile adoption story from NHN," *Proceedings - International Computer Software and Applications Conference*, pp. 476–481, 2012.

- [22] D. Bustard, "Beyond mainstream adoption: From agile software development to agile organizational change," *Proceedings -2012 IEEE 19th International Conference and Workshops on Engineering of Computer-Based Systems, ECBS 2012*, pp. 90– 97, 2012.
- [23] A. A. Lorber and K. D. Mish, "How We Successfully Adapted Agile for a Research-Heavy Engineering Software Team," in *2013 Agile Conference*, pp. 156–163, IEEE, aug 2013.
- [24] D. Bustard, G. Wilkie, and D. Greer, "The maturation of agile software development principles and practice: Observations on successive industrial studies in 2010 and 2012," *Proceedings of the International Symposium and Workshop on Engineering of Computer Based Systems*, pp. 139–146, 2013.
- [25] M. Drury, K. Conboy, and K. Power, "Obstacles to decision making in Agile software development teams," *Journal of Systems and Software*, vol. 85, no. 6, pp. 1239–1254, 2012.
- [26] T. J. Gandomani, H. Zulzalil, A. Azim, and A. Ghani, "How Human Aspects Impress Agile Software Development Transition and Adoption," *International Journal of Software Engineering and Its Applications*, vol. 8, no. 1, pp. 129–148, 2014.
- [27] C. de O. Melo, V. Santos, E. Katayama, H. Corbucci, R. Prikładnicki, A. Goldman, and F. Kon, "The evolution of agile software development in Brazil," *Journal of the Brazilian Computer Society*, vol. 19, no. 4, pp. 523–552, 2013.
- [28] M. Hummel, C. Rosenkranz, and R. Holten, "The Role of Communication in Agile Systems Development: An Analysis of the State of the Art," *Business and Information Systems Engineering*, vol. 5, no. 5, pp. 343–355, 2012.
- [29] T. Ihme, "Scrum adoption and architectural extensions in developing new service applications of large financial IT systems," *Journal of the Brazilian Computer Society*, vol. 19, no. 3, pp. 257–274, 2013.
- [30] K. Korhonen, "Evaluating the impact of an agile transformation: A longitudinal case study in a distributed context," *Software Quality Journal*, vol. 21, no. 4, pp. 599–624, 2013.
- [31] M. Pikkariainen, O. Salo, R. Kuusela, and P. Abrahamsson, "Strengths and barriers behind the successful agile deployment insights from the three software intensive companies in Finland," *Empirical Software Engineering*, vol. 17, no. 6, pp. 675– 702, 2012.
- [32] P. Bootla, O. Rojanapornpun, P. Mongkolnam, T. Dingsoyr, and T. Dyba, "Necessary Skills and Attitudes for Development Team Members in Scrum:" pp. 184–189, 2015.
- [33] V. G. Stray, Y. Lindsjorn, and D. I. K. Sjöberg, "Obstacles to efficient daily meetings in agile development projects: A case study," *International Symposium on Empirical Software Engineering and Measurement*, pp. 95–102, 2013.
- [34] D. Shahane, P. Jamsandekar, and D. Shahane, "Factors Influencing the Agile Methods in Practice-Literature Survey & Review," pp. 556–560, 2014.
- [35] M. Sihuy, A. Dávila, M. Pessoa, E. D. Posgrado, U. Nacional, and M. D. San, "Factores en la Adopción de Métodos Ágiles en el Proceso de Desarrollo de Software: Revisión Sistemática de la Literatura,"

- [36] T. J. Gandomani and M. Z. Nafchi, "An empirically-developed framework for Agile transition and adoption: A Grounded Theory approach," *Journal of Systems and Software*, vol. 107, pp. 204–219, 2015.
- [37] S. Nerur, R. Mahapatra, and G. Mangalaraj, "Challenges of Migrating to Agile Methodologies", *COMMUNICATIONS OF THE ACM* May 2005/Vol. 48, No. 5, pp. 73-78
- [38] J. Freeman. Common Problems Experienced When Adopting Agile Development, on line August 5th, 2015, retrieved from: <http://www.seguetech.com/blog/2015/08/05/CommonProblemsadoptingagiledevelopment>, on January 31, 2016.
- [39] K. Jammalamadaka, R. Krishna, "AGILE SOFTWARE DEVELOPMENT AND CHALLENGES", *International Journal of Research in Engineering and Technology*, Volume: 02, Issue: 08, Aug-2013, pp. 125-129.
- [40] A. Mahanti, Challenges in Enterprise Adoption of Agile Methods – A Survey, *Journal of Computing and Information Technology - CIT* 14, 2006, 3, 197-206.
- [41] R. Juárez-Ramírez, G. Licea, I. Barriba, V. Izquierdo, A. Angeles. (2011). "Engineering the development process for user interfaces: Toward improving usability of Mobile applications"; In *Communications in Computer and Information Science* Vol. 167, Springer. ISBN 978-3-642-22026-5, pp. 65-79. DOI: 10.1007/978-3-642-22027-2.
- [42] R. Popli and N. Chauhan, "A sprint-point based estimation technique in scrum," *Proceedings of the 2013 International Conference on Information Systems and Computer Networks*, ISCON 2013, pp. 98–103, 2013.
- [43] R Popli and N Chauhan. Cost and effort estimation in agile software development. Optimization, Reliability, and Information Technology (ICROIT), 2014 International Conference on, pages 57–61, 2014.
- [44] V. Lenarduzzi, "Could social factors influence the effort software estimation?" *Proceedings of the 7th International Workshop on Social Software Engineering - SSE* 2015, no. May, pp. 21–24, 2015.
- [45] H. Zahraoui, M. Abdou, and J. Idrissi, "Adjusting Story Points Calculation in Scrum Effort & Time Estimation," 2015.
- [46] N. C. Haugen, "An empirical study of using planning poker for user story estimation," *Proceedings - AGILE Conference*, 2006, vol. 2006, pp. 23–31, 2006
- [47] Florian Raith, Ingo Richter, Robert Lindermeier, and Gudrun Klinker. Identification of Inaccurate Effort Estimates in Agile Software Development. 2013 20th Asia-Pacific Software Engineering Conference (APSEC), pages 67–72, 2013.
- [48] Viljan Mahnic and Tomaz Hovelja. On using planning poker for estimating user stories. *Journal of Systems and Software*, 85(9):2086– 2095, 2012.
- [49] Kjetil Moløkken-Østfold, Nils Christian Haugen, and Hans Christian Benestad. Using planning poker for combining expert estimates in software projects. *Journal of Systems and Software*, 81(12):2106–2117, 2008.

- [50] B. Cartaxo, A. Araujo, A. S. Barreto, and S. Soares, "The Impact of Scrum on Customer Satisfaction: An Empirical Study," in 2013 27th Brazilian Symposium on Software Engineering, pp. 129–136, IEEE, oct 2013.
- [51] E. Ungan, N. Cizmeli, and O. Demirors, "Comparison of Functional Size Based Estimation and Story Points, Based on Effort Estimation Effectiveness in SCRUM Projects," 2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications, pp. 77–80, 2014
- [52] A. M. M. Hamed and H. Abushama, "Popular agile approaches in software development: Review and analysis," Proceedings - 2013 International Conference on Computer, Electrical and Electronics Engineering: 'Research Makes a Difference', ICCEEE 2013, pp. 160–166, 2013.
- [53] C. J. Torrecilla-Salinas, J. Sedeño, M. J. Escalona, and M. Mejías, "Estimating, planning and managing Agile Web development projects under a value-based perspective," Information and Software Technology, vol. 61, pp. 124–144, 2015.
- [54] Mike Cohn. Techniques for Estimating. Agile Estimating and Planning, pages 49–60, 2005.
- [55] D. Maximini, "The Scrum Culture," pp. 173–180, 2015.
- [56] J. López-Martínez, R. Juárez-Ramírez, C. Huertas, S. Jiménez, and C. Guerra-García, "Problems in the Adoption of Agile-Scrum Methodologies: A Systematic Literature Review," 2016.
- [57] M. Kohlbacher, E. Stelzmann, and S. Maierhofer, "Do agile software development practices increase customer satisfaction in Systems Engineering projects?" Systems Conference (SysCon), 2011 IEEE International, pp. 168–172, 2011.
- [58] Inayat, I., et al. A systematic literature review on agile requirements engineering practices and challenges. Computers in Human Behavior (2014), <http://dx.doi.org/10.1016/j.chb.2014.10.046>
- [59] A. Kanane, "Challenges related to the adoption of Scrum," pp. 1–31, 2014.
- [60] E. Cardozo, B. Araújo Neto, A. Barza, C. França, and F. da Silva, "SCRUM and Productivity in Software Projects: A Systematic Literature Review," 14th International Conference on Evaluation and Assessment in Software Engineering (EASE), pp. 1–4, 2010.
- [61] M. Kaisti, V. Rantala, T. Mujunen, S. Hyrynsalmi, K. Könnölä, T. Mäkilä, and T. Lehtonen, "Agile methods for embedded systems development - a literature review and a mapping study," EURASIP Journal on Embedded Systems, vol. 2013, no. 1, p. 15, 2013.
- [62] T. Dingsoyr, S. Nerur, V. Balijepally, N. Moe. A decade of agile methodologies: Towards explaining agile software development / The Journal of Systems and Software 85 (2012) 1213– 1221.

# Chapter # 22

## An Instructional Proposal for study of concepts on Software Engineering assisted by Ludic Virtual Learning Environments

**Raúl A. Aguilar, Juan P. Ucán, Julio C. Díaz y Antonio A. Aguilera**

Facultad de Matemáticas,  
Universidad Autónoma de Yucatán  
Mérida, México  
{avera, juan.ucan, julio.diaz, aaguilet}@correo.uady.mx

### 1. Introduction

Advances in Technology and Instruction have enabled to diversify Education Support Systems, initially, developing of this kind of system adopted the Computer Aided Instruction paradigm and subsequently it was refined with Artificial Intelligence techniques implemented in the Computer Aided Intelligent Instruction paradigm [1]. In recent decades, one of the lines of current research in the field of Computer Education revolves around the so-called Learning Environments, also known as Virtual Learning Environments [2], which are conceived as computer systems designed on purpose as spaces rich in situations that should promote meaningful learning in students.

In the instructional scope, a strategy that we find interesting to integrate to promote motivation among learners is the incorporation of ludic sceneries, also known as educational games. According to [3] the use of educational games as learning tools is a promising approach due to their abilities to teach and reinforce not only knowledge but also important skills such as problem-solving, collaboration, and communication.

In this chapter we describe an Instructional Proposal assisted by Ludic Virtual Learning Environments (VLE) that use of a competitive goal structure [4] in which a student —while playing— reviews concepts [5] associated with a course of Software Engineering.

The following section presents a brief reflection on Education in Software Engineering and educational program used as a reference in this work. The third section describes the performance of MemoSoft, VLE developed for use with students from educational programs in the area of Software Engineering; rules and dynamics of the instructional ludic proposal are also described in that section. The fourth section, the software development methodology used —based in UWE— to implement prototypes VLE is presented. In section five, describes in detail the pilot test with students as part of the process of empirical validation of our proposal. Finally, the authors' conclusions about the instructional proposal to use a Ludic VLE reviewing concepts in the area of Software Engineering are presented, as well as thanks to all who contributed to the development of the proposal is reported in this chapter.

## 2. Education on software engineering

Software Engineering (SE) is a professional discipline that will meet the next year just half a century of existence. In education, the development of the discipline began in 1978 with graduate programs in the United States, and in 1987 with programs in the undergraduate level in the UK. In the case of Mexico, the programs in the area of Software Engineering not see the light in the last century; incidentally is 2004 the year in which they begin to offer both the first master's program at the Center for Research in Applied Mathematics (CIMAT, for its acronym in Spanish), and the first undergraduate program offered in Mexico [6].

Education on Software Engineering has been influenced by entities such as the Institute of Electrical and Electronic Engineers (IEEE) who published in 2004 and updated in 2014, the Guide to the Software Engineering Body of Knowledge —called SWEBOK— document that provides a consensually validated characterization of the bounds of the

Software Engineering discipline and to provide a topical access to the Body of Knowledge supporting that discipline [7]. Association for Computing Machinery (ACM) is another entity, which together with the IEEE-Computer Society published 2004 and updated in 2015, Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering, which can serve as a reference for Educational Institutions and Accreditation Councils [8].

In Mexico, the National Association of Computer Education Institutions (ANIEI, for its acronym in Spanish) established in October 1982, in the absence of a core of knowledge for a professional Computer, it started working since 1983 on a set of Curricular Model Higher Level of Informatics and Computing; this proposal has four professional profiles in Informatics and Computing, among which is the profile of Software Engineer [9].

In our proposal the VLE uses as a reference the curriculum of the first Undergraduate Degree Program in Software Engineering offered in Mexico; It should be noted that this program has been nationally recognized for its educational quality , being accredited in 2013 by the National Accreditation Council in Informatics and Computing (CONAIC, for its acronym in Spanish), as well as the recognition received by the National Assessment Center (CENEVAL, for its acronym in Spanish) to enter in 2014 to the Registry of Programs Academic Degree High Performance - EGEL , and endorse its membership in 2015. Figure 1 illustrates the list of subjects considered in the VLE.

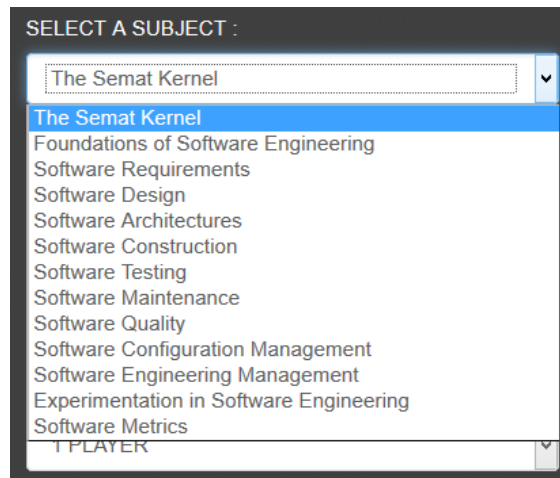


Figure 1. List of subjects considered in the VLE.

### 3. Memosoft: A Ludic VLE

MemoSoft is a Ludic VLE which aims to assist students in reviewing concepts related to Software Engineering topics. The instructional proposal to carry out an adaptation of the popular game “Memorama” in which a set of pairs of hidden images are distributed on a game table; players must, at every turn, be revealing pairs of cards and memorizing the position in the table images of both the player displays, like his opponent; each player keeps his turn and accumulate points to be revealing pairs of matching images.

The second prototype MemoSoft was implemented based on the requirements specification of the first prototype described in [10] and incorporated additional requirements, such as responsive design [11] which was built to be adaptable to any electronic device —telephone phone, tablet, etc. Figure 2 illustrates the initial screen of MemoSoft 2.0 run from a tablet.



Figure 2. Inicial Screen MemoSoft 2.0

We have adapted Memorama dynamics. In this popular game, the game begins turning over couple of cards —with the same figure — and deals with the figure down randomly so it is not possible to see the image that is in them; a player must find pairs of images, revealing pairs of cards each turn and memorizing the position in the table of each of the images. With MemoSoft, players —students really— must uncover couples Concept - Text description of certain course related to Software Engineering. Table 1 illustrates an example of couples designed for an optional course called “The Semat Kernel” [12].

Table 1. Couples Concept-Description for a Subject

Concept	Description
Alphas	Representations of the essential things to work with.
Activity Spaces	Representations of the essential things to do.
Customer	Area of concern the team needs to understand the stakeholders and the opportunity to be addressed:
Endeavor	Area of concern the team and its way-of-working have to be formed and the work has to be done.
Stakeholders (Alpha)	The people, groups, or organizations who affect or are affected by a software system.
Requirements (Alpha)	What the software system must do to address the opportunity and satisfy the stakeholders.
Stakeholder Representation	This competency encapsulates the ability to gather, communicate and balance the needs of other stakeholders, and accurately represent their views.
Development	This competency encapsulates the ability to design and program effective software systems following the standards and norms agreed upon by the team.
Testing	This competency encapsulates the ability to test a system, verifying that it is usable and that it meets the requirements.
Leadership	This competency enable a person to inspire and motivate a group of people to achieve a successful conclusion to their work and to meet their objectives.



In a ring game, students should not only memorize the position of cards but also link the descriptions with the disclosed concepts, such a dynamic opportunity to reaffirm the theory studied during class sessions. According to the specifications defined for MemoSoft, at the beginning of the game, a trainee must select the theme, the level domain, and the number of players; in the case of the second prototype —implemented for individual assessment by a group of students— the game it was configured for single player (see Figure 3). With respect to the domain for the game, MemoSoft 2.0 considers three levels of difficulty, and associated with a limited period of player participation in a session time:

- Easy (400 seconds),
- Medium (300 seconds),
- Hard (200 seconds).



Figure 3. Game setup screen.

At the start of a game the position of each of the letters of concepts (blue) and descriptions (pink) is assigned randomly by the VLE and placed face down (hidden); player timer is initialized according to the selected level. Figure 4 illustrates the beginning of a game level—at de medium level—with the nine hidden pairs in the MemoSoft 2.0. It is noteworthy that another of the improvements implemented in this second prototype was the reduction in the number of couples considered in a ring game in MemoSoft 1.0 [10], the number of pairs is ten. However, to increase the size of the letters and typography of texts as well as to eliminate the possibility of a tie between players — at having an even number of cards in the game—it was decided to reduce the number of cards couple (see Figure 4).



Figure 4. View of the initial state of a game MemoSoft Ver. 2.0

On each turn, the student can turn a couple of cards at a time, and to decide if the cards are linked disclosed, according to your choice, four events are triggered as shown below

(1) Pair of right cards and the player chooses "Pair"

→ Collect three points,

(2) Pair of right cards and the player chooses "Not Pair"

→ Are subtracted three points,

(3) Pair of incorrect cards and the player chooses "Pair"

→ Are subtracted 3 points,

(4) Pair of incorrect cards and the player chooses "Not Pair"

→ Collect one point.

Note: Note that in (1) the player gets a new turn.

In the case of a game with two players, as has been implemented in Memosoft 1.0, a course of events between two players could be illustrated in the following case (see Figure 5):

## Case 1

(1) The player #1 has -1 point earned for 9 innings, and the combination was:

He found three pairs = 9 points.

I was wrong 3 times [PAIR pressing the button] = -9 points

I was wrong 1 time [NOT PAIR pressing the button] = -3 points

He found two letters that were not couples = 2 points

(2) Player #2 has -8 points that has accumulated in 6 innings, the combination was:

Found 1 pair = 3 points.

I was wrong 4 times = - 12 points

1 couple found they were not partners = 1 point



Figure 5. Case 1 in MemoSoft 1.0

To conclude a two-player game in Memosoft, there are three completion events:

- Final 1. None was time runs out and the first and second place winner are determined depending on the score.
- Final 2. That the # 1 player runs out of time and the player 2 you are not exhausted and find the missing couple. It is determined the first and second place winner based on the score.

- Final 3. Both players are timing out, and determines the first and second place winner based on the score.

In the event that a student is exercised individually —as in the case of the pilot test in MemoSoft 2.0— the result is reported based on the number of pairs found and the number of accumulated points and have two completion events:

- Final 1. That the player is timing out and has not found all the missing couples; it tells you your score and number of couples found.
- Final 2. The player discovers all couples; the player has won (see Figure 6).



Figure 6. Final 2 in MemoSoft 2.0

MemoSoft 2.0 also provides a User Manager, with which the actions of registration, deletion and modification of users (Figure 7) are made.

MANEJADOR DE CONCEPTOS   MANEJADOR DE USUARIOS   MANEJADOR DE MATERIAS   SALIR						
id	nombre	contraseña	Tipo	Editar		Eliminar
1	administrador01	*****	administrador	Actualizar		Eliminar
2	antonio alonzo	*****	usuario	Actualizar		Eliminar
3	miguel burgos	*****	usuario	Actualizar		Eliminar
4	juli burgos	*****	usuario	Actualizar		Eliminar
5	andre chay	*****	usuario	Actualizar		Eliminar
6	felix diaz	*****	usuario	Actualizar		Eliminar
7	roberto franco	*****	usuario	Actualizar		Eliminar

Figure 7. User Manager Vier in MemoSoft 2.0

On the other hand, for the registration of new couples concept-description associated with new subjects, MemoSoft 2.0 has Concepts Manager which can be captured each pair individually, or loaded via an Excel file (see Figure 8).

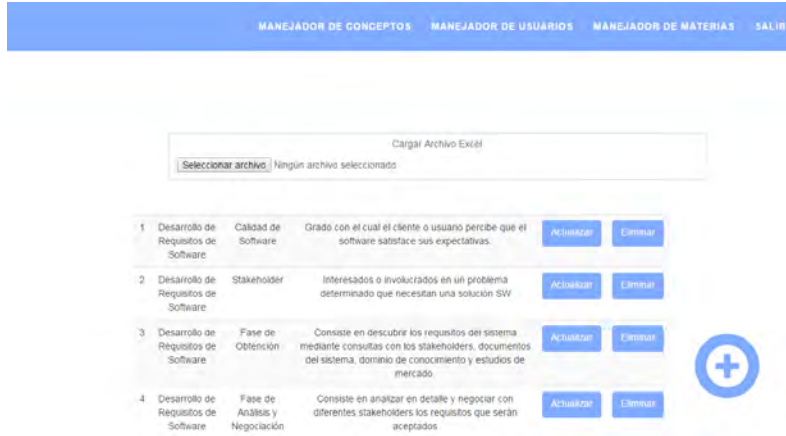


Figure 8. Concepts Manager View in MemoSoft 2.0

## 4. VLE Development Using Web Engineering

MemoSoft is a hypertext application designed to be used individually or collectively via Internet. By its nature, MemoSoft was developed using Web Engineering, i.e. applying systematic, disciplined and quantifiable methodologies in the efficient development, operation and development of high-quality applications on the World Wide Web [11]. For this purpose, UML-based Web Engineering (UWE) technology was selected. UWE proposes the development of five models [12]:

- Requirements Model. Documents functional requirements of the Web application through a Use Case Model.
- Content Model. Define, by a class diagram, the concepts involved in the application.
- Navigation Model. It is used to represent navigation within the application and a set of structures such as indexes, menus and queries.
- Presentation Model. Provides an abstract view of the user interface of a web application.
- Process Model. Represents the appearance of activities that connect with every kind of process.

Due to limitations on the extension of this work, in this section only three of these models are presented: Content Model, which corresponds to a traditional Object Oriented Model that describes the entities using UML notation, the Presentation Model, in which interface objects is described, and Process Model.

## 4.1 Content Model

Content Model corresponds to a traditional Object Oriented Model and provides a visual specification of the information in the primary domain of the web application. In Figure 9, the class diagram for the Model Content is presented.

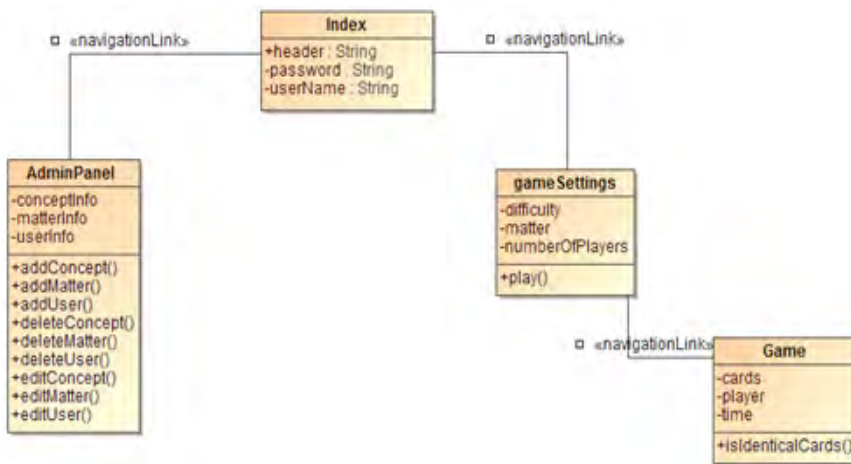


Figure 9. Content Model

As can be seen, the Content Model defines the following classes: Index class contains data for user registration; AdminPanel class contains information about the game subject; the gameSettings class contains the configuration of a game, such as difficulty, subject and number of players, and Game class contains game information such as cards, player and the elapsed time for the game.

## 4.2 Presentation Model

Presentation Model provides both Navigation and Process classes not displayed in the Navigation Model and belong to this Web application. Presentation Model describes the basic elements of the user interface, i.e., text, images, links and forms, and others.

This model facilitates the design of the elements in terms of their position and function before construction. Figure 10 illustrates the modeling of the MemoSoft main page (Index).

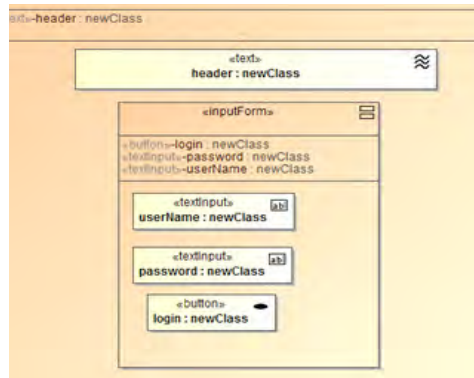


Figure 10. Presentation Page: Index

Main Page, as displayed in Figure 10, is modeled on a “presentationPage”, texts are modeled using “text”, and “bottom” for sending the “inputForm”. In Figure 11, configuration page for a gameplay is modeled.



Figure 11. Configuración de una partida

In modeling the configuration page, a template has been integrated, where topic, difficulty and number of players can be selected, using «selection» class, and «button» class for the submit button; at the top remains the header and also it integrates an «imageInput» class for the figures displayed in the game. Figure 12 shows the page where the

game is displayed. The stereotype “customComponents” organizes, at the right side, the game cards distribution; on the left side of the page the player’s name, time and score are displayed, and they are modeled with the stereotype «text».

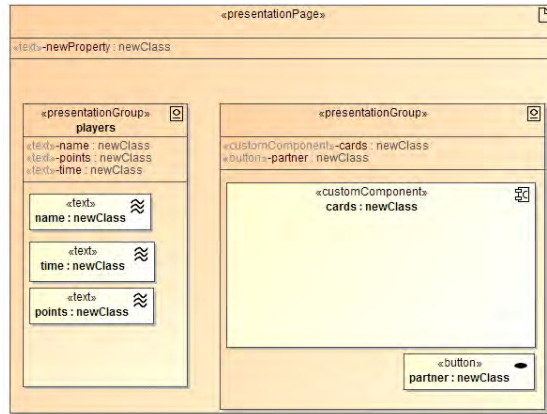


Figure 12. Página de juego o partida

### 4.3 Process Model

The MemoSoft navigation can be described by processes classes that represent the input and output of business processes. In the process model, both process structure and the flow of activities connected with each «processClass» are modeled. Figure 13 illustrates the activities of a game session with MemoSoft. Once the player is registered, sequential processes are choice of subject, choice of difficulty level and subsequently the number of players; from this point, the game starts.

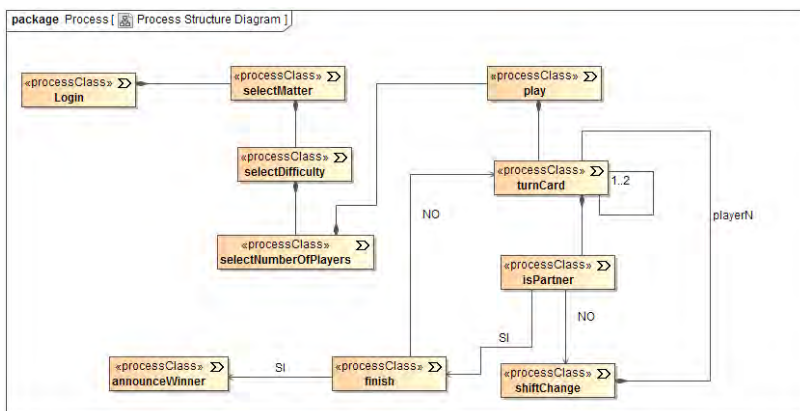


Figure 13. Modelo de procesos



## 4.4 Technology Used in MemoSoft

For the first MemoSoft prototype, we decided to use the AJAX technology (see Figure 14). AJAX uses the JavaScript XMLHttpRequest object for asynchronous requests to the server; that is, requests which do not require updating or refreshing the whole page. The advantages of using XMLHttpRequest for Web applications are 1) send or receive information from or to the server, 2) ask the server to perform operations, and 3) change the appearance of the current Web page. The above, without the user having to be redirected to another page, and without changing the element that has the focus [15].

In the development of the prototype it has been used PHP, and data persistence model uses MySQL. Also, in the client-side, one of the frameworks with AJAX JQuery UI is implemented [16].

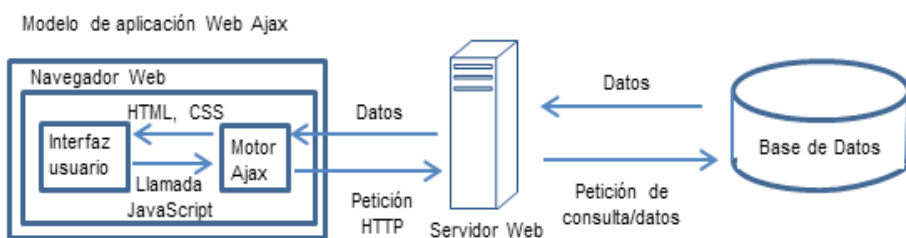


Figure 14. Ajax Architecture used in MemoSoft

Notably, all the technologies used for the development of MemoSoft, are open source, as one of the practices undertaken by our research group for software development.

## 5. Empiric Validation

For empirical validating of the instructional proposal, a pilot test with the students was planned, following some of the principles, which guide its development [17].

### 5.1 Goals

The first goal of the test consisted on validating the fulfilment of technical and pedagogical aspects [18]. Table 2 lists the factors considered in the evaluation.

Table 2. Elements considered in the empiric validation

Aspects	Factors
Technical	* Usability
	* Correctness
Pedagogical	* Instruction
	* Didactic Material

In order to obtain an assessment of the factors considered in the test, an instrument was designed consisting of eight items of structured response and the second section with three items of non-structured response. Figure 15 shows the first section of the instrument.

Research Group of Technologies for Education on Software Engineering

**PROJECT “EDUCATION ON SE USING GAMIFICATION”**  
**(Instrument for Empiric Validation of MemoSoft)**

Student: \_\_\_\_\_

<b>Instructions:</b> For each assertion, mark with an X the square that corresponds with your degree of agreement.		<b>TA</b> Total Agreement <b>AG</b> Agreement <b>NA</b> It does Not Apply <b>DI</b> Disagreement <b>TD</b> Total Disagreement				
		TA	AG	NA	DI	TD
01	The required effort to learn to use the environment has been minimum.					
02	The time available in the environment for a game session is appropriate, in accordance with the selected difficulty level					
03	The size of the letters is appropriate in interaction with the gaming environment					
04	The number of displayed cards is suitable for interaction with the environment					
05	The size and font in the cards is suitable for interaction with the environment.					
06	The descriptions used for concepts are clear.					
07	The dynamics of the game with MemoSoft, is an appropriate mechanism for the exercise of concepts in the subject.					
08	The dynamics of the game with MemoSoft, is an appropriate mechanism for assessing learning concepts related to the subject.					

Figure 15. The instrument of evaluation (structured response section).

Factors considered about the technical aspects were:

- Usability. Facility with which a user makes use of the environment according to the level of previous preparation (item: 01).
- Correctness. The capacity of the environment to respond to the requirements for which it has been developed according to the difficulty level selected (item: 02).

The second aspect considered in the evaluation is related to pedagogical factors:

- Didactic material. Static (texts, images) and dynamic (sequences, simulations, etc.) means used to support the instruction process (items: 03, 04, 05 & 06).
- Instruction. The process by means of which a human tutor promotes the learning (items: 07 & 08).

The section of free opinions (non-structured response) was used to obtain additional information about the two aspects mentioned before.

The second goal was to identify the strategies used by students during the session; this second goal was established with the intention of identifying possible changes in the rules of the game.

## 5.2 Preparation

The VLE target population are students in career courses in Software Engineering; in particular, MemoSoft was designed by reference to the Curriculum Program that operates at the Autonomous University of Yucatan (UADY, for its acronym in Spanish).

To carry out the pilot test, one of the groups that studied the educational program in January-May 2016 period was selected; in particular, the group of students enrolled in the course Software Development Requirements.

In order to have a greater number of subjects of analysis, the use of VLE individually, as well as the possibility that students will use their own equipment during exercise was considered. For this purpose, during the period from January to April, as a final project of the course called “Web Engineering”, a group of students the same educational program implemented the second prototype of MemoSoft with features that have been described in previous sections and incorporated a set of couples concept-description related to the topics covered on the subject throughout the course. The prototype was revised

and released by the research group and proceeded to request authorization for installation on one of the servers of the institution; finally, MemoSoft 2.0 was available for use through the network of the institution from the third week of May.

Note that the teaching of the subject did not undergo modification during the course, and was implemented according to the initial educational planning.

The pilot test of MemoSoft with students, initially was conceived as part of the dynamics of one of the class sessions; however, one of the recommendations of the audience during the presentation of the work in the CONISOFT [10], was the MemoSoft use in an evaluation activity, which we found the activity quite interesting to implement; once discussed, the research group decided to incorporate this activity in the latest performance test scheduled at the end of the course. According to educational planning, the second performance test corresponds to units 5, 6, 7 and 8 was scheduled for May 20, 2016.

The physical space for conducting the test, a room computer center with capacity for 30 students was used, and functionality of the equipment was verified prior to test execution.

In the case of the assessment, a two section instrument was designed: the first section with five items unstructured response to the issues discussed in class, and a second section consisted of the use of MemoSoft in two sessions of play. The second section was also an opportunity to earn additional points to the value of the test; this mechanism was incorporated as an aspect of extrinsic motivation for the game, which has been reported among good practices in designs using the theory of Gamification [19].

### 5.3 Execution

The pilot test implementation was conducted in an evaluation session of the course Software Development Requirements; performance test for students was designed to be completed in a maximum time of 120 minutes, and the session was organized as follows:

- Reading and clarification of doubts about assessment tool, by the teacher, (5'-10').
- Resolution in writing of Section I of the assessment instrument (70'- 80').
- Using computer equipment, and following the instructions in section II of the instrument, complete two sessions of play with MemoSoft (12'-15').
- Responding to an opinion poll (10'- 15').

The number of students enrolled in the course was 23, however, three students, for various reasons, did not attend the evaluation session, so that the sample consisted of 20 male students —the only female student enrolled in the group did not attend for initiating a research stay in Canada.

It is noteworthy that the dynamics of the performance test for the second section of the evaluation instrument —the game with MemoSoft— was described at the time of the evaluation session, so students had not used the VLE previously.

## 5.4 Results

The results obtained with the implementation of the instrument were very favorable with respect to the technical aspects; with regard to usability, 95% of respondents expressed a favorable opinion, and as for the correction factor, 90% of respondents also expressed a favorable opinion.

Figure 16 shows a graph of the frequency of students' opinions obtained in each of the eight items structured response.

Regarding the aspect linked to instruction, 85% gave a favorable opinion on the use of VLE for exercise activities and 75% in evaluation activities; at this point, non-favorable opinions regarding its use as a mechanism for evaluating learning can be explained by the views expressed in response unstructured item, opinions were in the sense that the game can generate stress in the learner during the evaluation session, or the students could try to accumulate —according to their strategy— a positive score, rather than try to find couples.

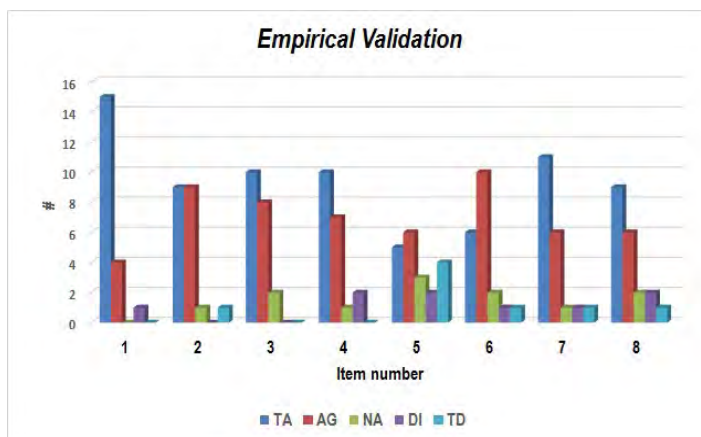


Figure 16. Opinions collected during the pilot test

In the aspect related to the didactic material, the item that had greater dispersion in the sense of the opinions —55% favorable, 30% unfavorable and 15% without opinion— was related to the size and font used, in this sense, although the second prototype was implemented just thinking of a responsive design, it is necessary a new review for the implementation of this aspect, especially considering the diversity of artifacts interaction (Screen, Tablet, Mobile).

Although the other three items that gathered opinions to validate the pedagogical aspect, factor related teaching materials received favorable opinions —on average 80%— the students expressed in item response unstructured confirm the need to review certain aspects of interface design; some of the comments were not fully responsive, on small monitors the cards do not appear completely, it is preferable to have pop-up messages instead of using a section of the screen.

As for the second objective, 90% said they did not use a defined strategy for identifying couples, rather unveiled cards trying to identify couples via trial and error. As for the remaining 10% said that they tried to discard the erroneous couples, memorizing the positions of the already disclosed. Interestingly expressed was about the order in which the students selected the cards, 15% said that first unveiled a concept card and then a description card.

## 6. Conclusions

We have proposed the use of a Ludic Virtual Learning Environment for the exercise of concepts related to the area of Software Engineering. Two prototypes were developed and MemoSoft 2.0 was subjected to a test pilot with students in an evaluation activity to validate empirically the features and rules set out therein. The opinions related aspects of usability and instruction were generally positive; however, opinions linked with teaching materials, induce us to review the features of the environment. Regarding the strategies used by students during the game individually, no specific patterns observed. In the other hand, the use of MemosSoft as a tool to assist students in their learning process, apparently it has resulted, with the information obtained in the pilot test, a suitable option that can be incorporated into learning activities, both presential modality as in mixed modality. With the above results, it is possible to direct our efforts to modify certain aspects of the interface and enable the collective game, to analyze strategies in competition scenarios with peers.

Lessons learned with developing the proposal allow us to suggest that its use can be extended not only to activities for the review of concepts, but also evaluation activities.

## 7. Acknowledgements

Authors acknowledge the support received from the administration of the Faculty of Mathematics (UADY, for its acronym in Spanish) for the financial support through the project PROFOCIE 2015-31-12 for reporting of progress in CONISOFT 2016 held in the city of Puebla. Also, special thanks to the students: Andre Chay, Edwin Gonzalez, Jonathan Gonzalez and Eduardo Rodriguez, for their support in the programming —as part of its project in the course Engineering Web— of the second prototype for MemoSoft. Finally, we thank the group of students who were studying the subject Software Requirements Development, the valuable feedback we received regarding the Ludic VLE as part of the process of empirical validation.

## 8. References

- [1] W. Clancey and E. Soloway. “Artificial Intelligence and Learning Environments” A Special Issue of Artificial Intelligence: An International Journal. 1990.
- [2] P. Dillenbourg, “Virtual Learning Environments” Workshop on Virtual Learning Environments, 2000.
- [3] D. Dicheva, C. Dichev, G. Agre and G. Angelova, “Gamification in Education: A Systematic Mapping Study”. *Educational Technology & Society*, 18 (3), 2015
- [4] D. Johnson and R. Johnson, “Learning together and alone: Cooperative, Competitive, and Individualistic learning,” Englewood Cliffs, NJ: Prentice Hall. 1994.
- [5] G. Kearsley, “Artificial Intelligence & Instruction. Applications and Methods” Addison-Wesley, 1987.
- [6] R. Aguilar, y J. Diaz. “La Ingeniería de Software en México: hacia la consolidación del primer programa de licenciatura”. *Revista de Tecnología Educativa*. Vol 2. Num. 2. Pp. 6-17. 2015.
- [7] P. Bourque and R. Fairley. “Guide to the Software Engineering Body of Knowledge (SWEBOK V3.0)”, IEEE Computer Society, 2014.
- [8] ACM & IEEE-CS. “Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering (SE2014)”, 2015.
- [9] A., García, F. Álvarez, y M. Sánchez, M. (2015). *Modelos Curriculares del Nivel Superior de Informática y Computación*. Pearson.
- [10] R. Aguilar, I. Aké, J. Ucán, y A. Aguilera. “Desarrollando Entornos de Aprendizaje con UWE: Una aplicación lúdica para la ejercitación de conceptos en Ingeniería de Software”. *Memorias del 4to. Congreso Internacional de Investigación e Innovación en Ingeniería de Software*. pp.11-116. 2016.

- [11] J. Voutilainen, J. Salonen, & T. Mikkonen. "On the design of a responsive user interface for a multi-device web service". Proceedings of the Second ACM International Conference on Mobile Software Engineering and Systems (pp. 60-63). IEEE Press. 2015.
- [12] I. Jacobson, P. Wei Ng, P. McMahon, I. Spence, and S. Lidman, "The Essence of Software Engineering. Applying the SEMAT Kernel,". Addison-Wesley, 2013.
- [13] M. Gaedke and G. Gräf, "Development and Evolution of Web-Applications using the WebComposition Process Model," 2000.
- [14] N. Koch, A. Knapp, G. Zhang, and H. Baumeister, "Uml-Based Web Engineering," in Web Engineering: Modelling and Implementing Web Applications SE - 7, G. Rossi, O. Pastor, D. Schwabe, and L. Olsina, Eds. Springer London, 2008, pp. 157-191.
- [15] L. Balbin, "Beginning Ajax with PHP: From Novice to Professional," Apress, 2006.
- [16] jQuery user interface. <http://jqueryui.com/>. Accedido en enero del 2012.
- [17] A. Galvis. "Ingeniería de Software Educativo". Ediciones Uniandes, 1992.
- [18] P. Marqués. "Software Educativo. Guía de Uso y Metodología". Estel, 1995.
- [19] F. Llorens, F. Gallego, C. Villagrà, P. Compañ, R. Satorre, R. Molina. "Gamificación del Proceso de Aprendizaje: Lecciones Aprendidas" VAEP-RITA Vol. 4, Núm. 1, pp. 25-32. 2016.



# Chapter # 23

## Augmented Reality Applied in the Museum of Memory of Tlaxcala

**Marva-Angélica Mora-Lumbreras, Sergio Molina-Guarneros, Carolina-Rocío Sánchez-Pérez**  
Facultad de Ciencias Básicas, Ingeniería y Tecnología  
Universidad Autónoma de Tlaxcala  
Calzada Apizaquito s/n, CP 90401  
Tlaxcala, México.  
{marva.mora, sergio.molina.guarneros, krlinasp}@gmail.com

### 1. Introduction

Augmented Reality is an area that combines real elements with virtual elements in real time, where there is no substitution, but a complement of information. The additional information is images, 3D objects, videos, sounds and texts. Augmented reality is a relatively new area, dating from about the nineties [1, 2]. On the other hand, it is said that a museum is a public or private, permanent institution, with or without profit, serving the society, which acquires, conserves, researches, communicates and exposes or exhibits for purposes of study, education and delight art collections, scientific, etc., always with a cultural value, according to the International Council of Museums [3]. An excellent area that can be used favorably is the Augmented Reality that combines real aspects, such as pieces that has a museum, but also considers virtual aspects, which can be incorporated to provide real-time information on mobile devices of the users, such as video, audio descriptions, modeled three-dimensional, photo galleries, etc. Additional information allows introducing some level of interaction, without damaging the pieces. Currently, you can find a variety of projects that use digital technology in different fields, such as medicine, architecture, culture, videogames, etc. A clear example is the Casa Carranza museum [4], which includes 360 degrees views. The National Museum of Popular Culture [5] and the Louvre Museum [6] has online tours, the castle of Alcaudete and the Museum of Andalusia [7] have a circular structure with markers of Augmented Reality. Some museums that have incorporated Augmented Reality in mobile devices are the Museum of Modern Art and New York Museum. The Museum of London has an App where old photographs are merges with the current view. Finally,

we can mention the recreation of the Berlin Wall Museum [8], which results attractive in the tourism sector.

## 2. Augmented Reality in Mobile Devices

In recent years, the technological evolution is allowing an interesting transition in all areas; with the introduction of smart devices, the researchers have a great opportunity of developing extraordinary projects. Exploiting the mobile devices characteristics y the possibility of introducing Augmented Reality technology, this project presents a focused on a museum with the objective of improving the interaction, giving dynamism and additional information.

Augmented reality combines real information with virtual, creating a mixed reality in real time, therefore, adds a virtual synthetic part to reality, it considers geographical the user's position, markers, in addition, the mobile devices access online and stored information [9].

## 3. Generations of the museums

There are five generations of museums [10]:

- The first generation of museums is characterized as scientific and technical, it presents the classic concept of storing precious pieces, rare items, and masterpieces of nature or man.
- The second generation includes technological museums, based on the Industrial Revolution and the artisan classes.
- The third generation introduces the interaction, centering in the experienced visitor.
- The fourth generation can be identified as incorporating scientific theme parks. Its most prominent feature is the combination of information, education and entertainment in a product.
- Finally, the fifth generation uses the media, audiovisual show, special effects, surround sound techniques, virtual reality spaces resulting in scientific communication and education. Likewise, technology has led to the generation of fully virtual museums, where the physical museum is not relevant.

The memorial museum of Tlaxcala corresponds to the museums of the second generation, it has pieces of the founding of New Spain, agriculture tools in the sixteenth century

until the Baroque stage, however, by incorporating Augmented Reality, it museum could be considered of the third generation.

## 4. Methodology

In order to develop a software project, it is important to establish the methodology to be used and have the necessary permits from the relevant institutions. The development of a project Augmented Reality in museums includes people specialized in different areas, as anthropologists, historians, designers, programmers, etc. Specifically, this article focuses on software development. Although we searched different methodologies for this project, we decided to generate one, in Figure 1. The methodology defined for the project.

1. -Analysis of requirements	2. -Design and modeled	3. -Implementation	4. -Tests and Corrections:
<b>Goals:</b> A) Selection of museum areas B) Selection of museum pieces C) Selection of references <b>Techniques:</b> A) Mobile Augmented Reality B) Desktop Augmented Reality <b>Augmented Reality using:</b> A) Markers o images B) Position <b>Hardware</b> A) Developed B) User <b>Software</b> A) Developed B) Of application <b>Principles of Usability [8]</b> A) Interaction B) Presentation C) Navigation D) Panoramic E) Sound F) Orientation and help	<b>A) Design and modeled of software</b>  <b>B) Design of material:</b> Photographic Galleries , videos, text, Descriptive audios, 3D models, texturized, markers, images  <b>C) Interface Design</b>	<b>Codification</b>	<b>A) Unitary</b>  <b>B) Integration</b>  <b>C) Functional</b>  <b>D) Usability</b>  <b>E) Corrections</b>

Figure 1. Design Methodology for Implementation Augmented Reality focused Historical Museums

Detailing the methodology:

1. Analysis of requirements consider:
  - » Specific goals, in this case, selecting areas of the museum, key parts and literature to supplement the original information.
  - » Techniques to be used as Mobile Augmented Reality or Desktop
  - » Determine the use of markers or positions
  - » Definition of hardware of development and end-user
  - » Definition of software of development

- » Define principles of usability considering interaction, presentation, navigation, panoramic, sound, guidance and help.
- 2. Design and Modeled of software: UML, 3D objects, as well as taking into consideration usability aspects.
- 3. Implement
- 4. Finally, like all quality software is important to realize different tests as unitary, of integration, functional, of usability. Also is necessary correct each problem detected.

## 5. Selection of material to generate Augmented Reality

The Museum of Memory of Tlaxcala covers parts of the foundation of New Spain in the sixteenth century until the Baroque stage. This museum is divided into different thematic units:

- The republic of naturals
- The material world
- The economic life
- Devotions
- Memory
- The diaspora
- The heritage

Of this thematic units, we decided to work in all sections, but only 20 representative pieces of the museum, covering the following categories:

- Religious pieces
- Photo galleries of ranches
- Photo galleries of codices
- Tools of Agriculture

These 20 pieces were used for producing Augmented Reality, supplement the traditional information with technology. Figure 2 shows the modules of the project.

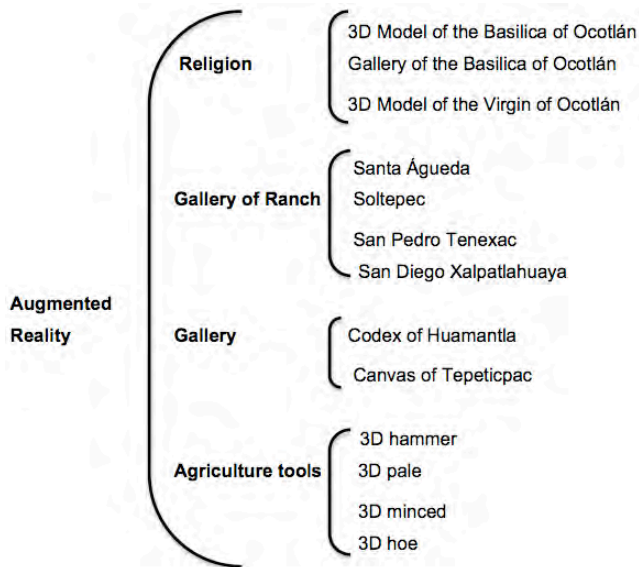


Figure 2. Modules of Augmented Reality focused on the Museum of Memory of Tlaxcala

## 6. Technical Specifications of Development

For the Augmented Reality project at the Museum of Memory of Tlaxcala was chosen as technical the Mobile Augmented Reality using markers, because the markers can be placed in key areas for an explanation in real time using mobile devices. The development was carried out in a laboratory of the Faculty of Basic Sciences, Engineering and Technology of the Autonomous University of Tlaxcala, which is equipped with computers for gamers Asus, con VCD ASUS Nvidia GTX 550 T1GB DDR5. While on the side of development, the software used is:

- The 3D graphics engine, created by Unity Technologies, founded in 2004 by David Helgason, Nicholas Francis and Joachim Ante in Copenhagen, Denmark [11]. 2. FrameMarker of Vuforia SDK developed by Qualcomm, a US company founded in 1985, according to Qualcomm Inc. Annual Report [12].
- Android developed for Android Inc., a company of software located in Palo Alto [13].

For the project had a high level of quality, design using the following usability principles [14]:

- Interaction is achieved by incorporating the use of markers, showing the user times and situations with high clarity on their mobile devices, with immediate actions executed by the user, such as the rotations and translations of the mobiles devices.

- **Presentation.** This project is easy to use, features a logo related to the museum, the user must only place the mobile device with the application running in front of a marker and the information is displayed immediately.
- **System Navegación.** This project has a set of markers belonging to different sections of the museum, as some markers lead to photo galleries, there is a continue navigation system.
- **Overview.** The 3D information is generated depending on the user position and movement, the visibility of three-dimensional objects change according to the position of the device, and orientation, but also it is possible rotate the 3D object using the touchscreen system, it also manages the concept of displaying multiple objects simultaneously.
- **Sound.** It is necessary synchronizes view and sound. For example, on using videos.
- **Guidance and assistance.** The project includes help for the novice user.

## 7. Museum of Memory of Tlaxcala with Augmented Reality

The project Museum of Memory of Tlaxcala with Augmented Reality is created according to the flow shown in Figure 3. We choose the real objects, which were captured with a camera for being modeled in 3D, the augmented reality is generated with the different 3D objects and markers, producing then the user can see the 3D models displayed on the mobile device.

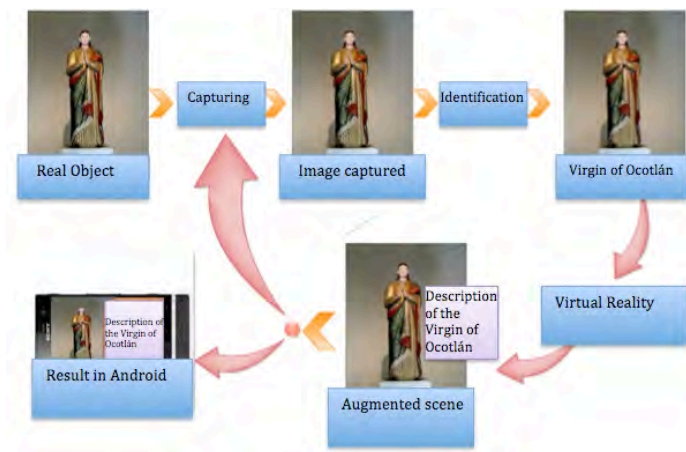


Figure 3. Flow of the project

The interface is intuitive for any type of person, the buttons representative actions, the project includes photo galleries, videos, texts and descriptive audio, 3D modeling, which allows visitors to have an interactive experience.

Specifically, this project includes a photo gallery and 3D-models of Ocotlan, different galleries of ranges such as Santa Agueda, Soltepec, San Pedro and San Diego Tenexac. Also, photo Galleries of the Codex of Huamantla and canvas of Tepeticpac agricultural tools modeladed in 3D. The implementation was carried out in one year.

Although the project includes several sections, we choose the San Pedro Tenexac ranges for this chapter, due to its significance. This ranch is located in Terrenate, Tlaxcala, the National Institute of Anthropology and History (INAH) declared Historic Monument of the Nation for their conservation in 1982. The gallery included photos of the San Pedro Tenexac ranch and details of the ranch. The user interaction is through of buttons that controlled the pass of the images on the screen. Figures 4 and 5 show the project operation. The information in Augmented Reality was obtained of historians, archeologists and references of this topic.

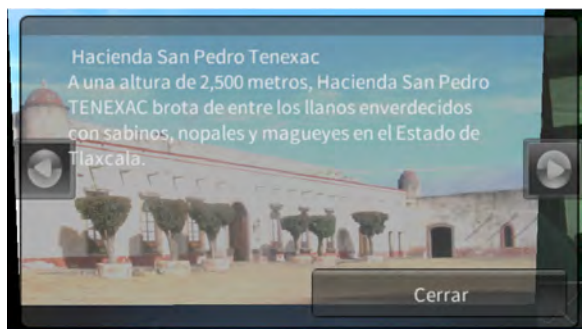


Figure 4. Textual description of the San Pedro Tenexac Ranch.



Figure 5. Tenexac Ranch.

Figure 6 shows how the application runs with some of the 3D models of agricultural tools



Figure 6. 3D models of agricultural tools.

The Basilica of Ocotlan is one of the main themes of the museum, so it is important to show the visitor how impressive the place, so we decided to show a 3D model.



Figure 7. 3D models of agricultural tools.

## 8. Usability Test

The case study was tested by unit, integration, functional and usability tests. The usability tests were done with users of museum visitors. After users installed the application they used to display the augmented information, then they answered two short surveys, the first focused on usability, the second for evaluating the selection of content and the user satisfaction. The tests were done to a group of 20 museum visitors with a variable range in age from 15 until 30 years; every visitor installed and used the application on a mobile device with Android. The application was provided by the team.

### Usability Survey

1. Was the project easy to install?
2. Was the project easy to explore?
3. Were the galleries easy to explore?
4. Is the user interface easy to understand?



Selection of content and user satisfaction.

1. Is the relevant information?
2. Does the additional information match with the real?
3. Is an innovative project?
4. Do you recommend the project?

Figures 8 and 9 present the results. It can be analyzed by the survey results, the project is simple to install, easily explore and easy to understand. By project content, users find it full, and most of them recommend it.

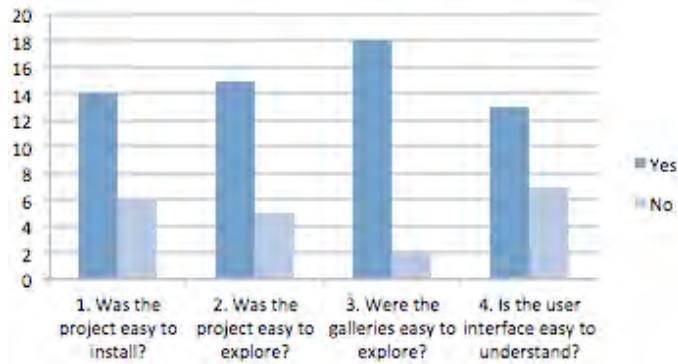


Figure 8. Results of the Usability

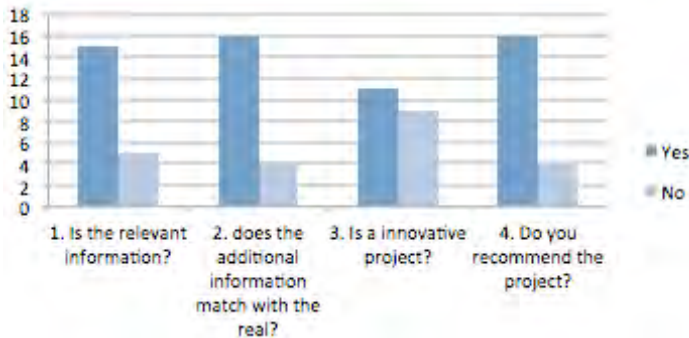


Figure 9. Results of the Content and User Satisfaction.

Visitors expressed that the inclusion of Augmented Reality is ideal, produce a good satisfaction in the visit of the museum. It is noteworthy that the application is not currently

available on any website or Google Play, due to it is in the final stage of testing. On the other hand, the university has massive visits of students and teachers from different schools and universities, in these visits, we have presented the project with 300 invited and 100 students of our university, giving a total of 400 testing within the University, all tests with positive comments.

## 9. Discussion

This chapter presents a study case for incorporating Augmented Reality at the Museum of Memory of Tlaxcala. For the development of this project, the cascade methodology was adapted. In this, we have incorporated key aspects for using Augmented Reality in museums of the first generation, we consider that with technology it museum could be considered of the third generation.

We observed during testing emotion and good satisfaction of the visitors, the interaction and participation justify suitability of this study, we could observe this as part of the tests that were performed. At the same time, it is important to note that all visitors saw with pleasure and satisfaction the incorporation of Augmented Reality in the traditional route, commented at the end of the evaluations that incorporate Augmented Reality in museums increase the attention, is useful and allows a degree of dynamism in an area where restrictions are very strong. Similarly, we observed in the massive evidence that the most students have mobile devices with Android.

On the other hand, if we compare the case study described here with some of the museums using technology, for example, Carranza House Museum [4] which contains 360 degrees views, photo galleries, online activities and Augmented Reality our project is different because it focuses on the use of virtual reality during the actual tour of the visitor, serving as a complement to the information shown, for example, the National museum of Popular Culture uses virtual tours online [5], which It makes it different to the here presented. We believe that our project enriches the information with videos, galleries, 3D objects as the Alcaudete Castle and the Museum of Andalusia [7] that use markers.

## 10. Conclusions

This project is evolving and was published in the fourth Edition of CONISOFT [14].

The technology is giving contributions in many areas, benefiting people to use in education and culture, enabling citizens to be better informed. However, even in today's

museums of Mexico, this effort is still inadequate, much work is required to improve the goals that technology imposes on us. This article presents a case study of Augmented Reality for the Museum of Memory of Tlaxcala, for the development of this project; a methodology that considers the main aspects to achieve quality software was adapted and presented. During testing case study could be seen that Augmented Reality is an area that handles interaction with visitors, this technology can complement a traditional exhibition and museum pieces are respected. Finally, noting that the technology is getting involved in all areas of the life, we can enjoy our culture respecting the history.

## 11. References

- [1] Barfield W, Caudell Thomas, *Fundamentals of Wearable Computers and Augmented Reality*, CRC Press. 2001
- [2] Tony Mullen, *Prototyping Augmented Reality*, JohnWiley & Sons, Inc. ISBN: 978-1-118-03663-1, 2011
- [3] Isabel R Villena, *Definitions of museums*. University of Castilla-La Mancha, 2009.
- [4] Museo Casa Carranza, conaculta. [http: //museocasadecarranza.gob.mx/multimedia/](http://museocasadecarranza.gob.mx/multimedia/), document retrieval date: January 8, 2016.
- [5] National Museum of Popular Culture. Moon metaphors: tradition and modernity in Indian art, Museo Nacional de Culturas Populares. [http: //museoculturaspopulares.gob.mx/multimedia/metáforas](http://museoculturaspopulares.gob.mx/multimedia/metáforas), document retrieval date: January 8, 2016.
- [6] Miyashita T, Meier P, Tachikawa T, Orlic S, Eble T, Scholz V, GAPEL A, Gerl O, Arnaudov S, Lieberknecht S, *An Augmented Reality museum guide*, 7th IEEE / ACM International Symposium on Mixed and Augmented Reality, 978-1-4244-2840-3, 2008, pp 103-106.
- [7] Ruíz Torres, *Augmented Reality and Museums*, Icon Magazine 14, 1697-8293, 2011, pp. 212-226.
- [8] Dolores Galindo, *museums Augmented Reality*, Social Museum, in July 2012
- [9] Fombona Cadavieco Javier Pascual Maria Angeles Sevillano, Madeira Ferreira Amador, *Augmented reality, an evolution of the application of mobile devices*, Pixel-Bit. Journal of Media and Education, (41): 197-210, 2012
- [10] E Antonio Ten Ros. *LOS scientific-technological. MUSEUMS A classification test for generations*. IEDHC (University of Valencia-CSIC)
- [11] Sue Blackman (2013), *Beginning 3D Game Development with Unity 4: All-in-one, multi-platform game development*, Ed TIA (Technology in Action).
- [12] Qualcomm, Inc. *Annual Report 2013*. Qualcomm Incorporated. Date Document Recovery November 18, 2015.
- [13] Hébuterne Sylvain, Pérochon Sébastien, *Android Application Development Guide for Smartphones and Tablets (2nd edition)*, Editions ENI, 2014. [14] Pérez Yulian-

ne, Castro Wilfredo, Garces Arianna, Usability Principles for virtual reality products, Spanish Academic Publishing, 2012.

- [14] Mora-Lumbreras Marva-Angélica, Molina-Guarneros Sergio, Sánchez-Pérez Carolina-Rocío, Augmented Reality: Case Study of the Museum of Memory of Tlaxcala, 4to. International Conference on Research and Innovation in Software Engineering 2016, CONISOFT'16.

# Chapter # 24

## Promoting Software Engineering Concepts to Children through a Serious Game

**Selene Ramírez-Rosales, Sodel Vázquez-Reyes, Juan Luis Villa-Cisneros, María de León-Sigg**  
Autonomous University of Zacatecas, Programa de Ingeniería de Software  
Ciudad Universitaria Siglo XXI, Edificio de Ingeniería de Software Ingeniería en Computación  
C.P. 98160, Zacatecas, Zac., México  
selenereamirezrosales@gmail.com, vazquezs@uaz.edu.mx, jlvilla@uaz.edu.mx, mleonsigg@uaz.edu.mx

### 1. Introduction

A serious, educative, or didactic game is one in which some type of knowledge is obtained through playing [1], facilitating the teaching and learning process [2] to solve problems or social challenges related to a subject [3]. Zyda [4] defines serious games as a “mental test, run through a computer that uses entertainment as a government or corporative training with educative, sanitary, public policy, and strategy communication goals”.

In addition, a serious game not only has an educational or serious purpose, but it also contains fun-focused elements [5]. Most important objectives in serious games are [6]:

1. Student motivation
2. Learning retention increase
3. Learning transfer improvement
4. Immediate feedback
5. Recognition obtaining

These objectives become attractive to increase and maintain the user attention, and because of these, serious games, are being described by some analysts as the next wave

of technology-mediated learning [7]. The uses for serious games include emergency services training, military training, corporate education, and even health care, making them useful without regard of objectives and level of the audience [7].

On the other hand, early learning of programming skills and technology-related subjects, improve school performance in other disciplines as mathematics and languages [8]. Due to this, is important to promote Software Engineering (SE) learning, as well as all of the software production aspects [9], including Object Oriented Programming (OOP), one of the most used programming paradigms [10]. Therefore, the goal intended with the research presented in this chapter was to develop a serious game to promote the interest about SE and OOP concepts to children older than eight years old, who use Android devices.

This chapter is organized as follows: In section two, is presented a brief review of projects to teach programming and SE concepts. The problem is described in section three; the serious game proposal is presented in section four. Later, section five explains how the serious game was developed. In section six is described the experimentation. The results of experiments are presented in section seven, and, finally, conclusions and future work are presented in section eight.

## 2. Related Work

Video games have been in our life for more than 50 years, becoming quickly in one of the most important, profitable and influential ways of entertainment in the United States and all over the world [11]. The term “serious game” was not used until 2002, when it was used for the first time during the launching of “America’s Army”, a military training and combat mission simulator, created by the American army and distributed for free over the Internet [12].

There are different serious projects to teach programming, oriented to children and young adults. Each project has characteristic properties that make them unique in their structure, rules, mechanisms, costs, platforms and potential user’s ages. Among the most important applications launched in recent years are Game Maker, Code.org, My Robot Friend, Scratch, Scratch JR, Cargo Bot, Code Hour, Code Monkey, Kodable Class, Tynker, Daisy the Dinosaur, Hopscotch, Lightbot, and Move the Turtle [13].

Those applications are oriented mainly to children between 5 and 17 years old, with activities according to those ages. Therefore, each one of those applications uses different styles and mechanics: drag and drop, puzzles, commands, challenges and tutorials.

Similarly, there are projects that support the learning of SE concepts, mainly focused on young adults, primarily university students. Those projects are usually simulators and products in a stand-alone platform. Notable examples are SimSE, SESAM, OSS, SimVBSE and The Incredible Manager [14, 15, 16, and 17].

In addition, most applications and projects mentioned before are free and/or accept donations to continue their development, and many are mobile and tablet applications. The most used operating system for them is iOS, even when Android is used in more than 50% of other operating systems.

However, even when those projects and efforts facilitate the promotion of SE and OOP, none of them groups together both areas of interest. Existent projects are simulations and practice games already in use in military, schools, and industry for learning. Those mainly facilitate the SE processes, allow virtual participation in realistic processes [11]

### 3. Problem

In a few years, the technological environment will be part of our life, and the knowledge about the use, functioning, development and maintenance of Information and Communication Technologies (ICT) will be essential [18]. However, elementary, middle and high school students and teachers, seem not to know about it, generating difficulties in the technological advance of society and economy [19].

Demand for programming knowledge is increasing, and next generations should know about it. Software development is going to be one of the most important fields of employment in the next ten years: there will be 1.4 million employments and only 400 thousand graduated professionals in the area [20].

Unfortunately, there are false stereotypes and myths about programmers and software engineers that have created the image of people who dedicate to do intimidating, challenging or boring activities [21]. Those false beliefs have affected children and young attitudes towards both subjects and have made difficult their training in SE and OOP.

### 4. Proposal

Currently, there are several projects promoting topics of SE and programming, but none of them is doing an integration of SE and OOP, considering as their specific audience scholar age children. Due to this, a serious game was developed to promote the interest

about SE and OOP for children older than eight years old [18]. The game works in Android because it is the most common platform for mobile devices.

Aspects covered in the serious game included how to solve programming problems [9], doing emphasis in the order:

1. Problem analysis
2. Requirements specification
3. Design
4. Implementation
5. Testing
6. Documenting
7. Maintenance

OOP, as one of the most used programming paradigms because of its simplicity to represent the way we perceive our environment, and its reusability, flexibility, scalability and understandability. OOP is divided into four main concepts that should be easily conceptualized by OOP paradigm users [10]:

1. Class: a data type that contains a single structure attribute and methods.
2. Object: a class instance
3. Attributes: class characteristics
4. Methods: class functions

OOP has another group of fundamental properties [10]:

1. Inheritance: objects are created from existent ones.
2. Polymorphism: objects respond in a different way to the same message.
3. Abstraction: focus on object's most important characteristics or facts.
4. Encapsulation: all needed elements are integrated into one entity.
5. Information hiding allows object attributes isolation or protection from the environment, in such a way that they cannot be modified in an arbitrary manner.



6. Modularity divides a large application into small parts, independent among them.

Other essential OOP concepts taken into account for the development of the serious game were:

1. Algorithms: a series of steps done to obtain a result from an input [22].
2. Conditional structures: if, if-else and switch
3. Iterative structures: while, do-while and for
4. Arrangements: data or element set of the same type and with the same identifier. Every element of the same arrangement is identified with the same name and the position or place it is located [10].

To conceptualize previous mentioned knowledge areas into a serious game, some elements and attributes should be considered [7, 23, and 24]:

*Dynamics*: elements that give game meaning. These dynamics are general aspects, objectives, effects, desires, and motivations (for example restrictions, emotions, narrative, progression, and relations).

*Mechanics*: this type of elements allows dynamics and game evolution. These are basic actions to motivate the user (for example challenges, opportunities, competition, cooperation, feedback, reward, victory states).

*Components*: this type of elements allows implementing game mechanics and dynamics (for example achievements, avatars, medals, content unblocking, scoreboards, levels, points, gifts, quests).

Based on game characteristics and objectives, the project was named Software KIDS. In Figure 1a y 1b, is shown logo, the name of the project and the characters that included in the serious game.



Figure 1a. Serious game logo, the name and characters



Figure 1b. Serious game logo, the name and characters

## 5. Development

As described in the previous section, Software KIDS is a serious game to promote SE and OOP with an Android device.

As software development is not an easy activity, numerous methodological proposals exist to solve different types of problems found. Because there is not a specific methodology to solve each of them, it is necessary to look for the most appropriate according to the characteristics of the problem [25]. Software industry frequently uses agile methodologies. Agile methodologies are based on four principles [25]:

1. Individuals and interactions over processes and tools.
2. Working software over comprehensive documentation.
3. Customer collaboration over contract negotiation.
4. Responding to change over following a plan.

One methodology that accomplishes such principles is SCRUMBAM.

SCRUMBAN is the combination of the best features and characteristics of agile methodology Scrum and Kanban. This is a combination of teamwork and working time optimization work [26]. The resulting combination provides the freedom to create unique solutions [26]. The details of both methodologies are shown in Table 1.

Table 1. Description of scrum and Kanban

Feature	SCRUM	Kanban
Time	Established time	Work In Progress
Iterations	Yes	No

This table continues on the following page————>

Backlogs	Sprint Backlog, Product Backlog	Backlog with limits
Roles	Yes	No
Restrictive	50%	0%
Changes	No	Yes
Estimations	Yes	No
Ceremonies	yes	No

SCRUMBAN was chosen because Software Kids development needed a way to maintain flexibility in project management. This was required because serious game development needs several iterations due to frequent changes [3]. Besides, there was no specific time limit and the working team was small.

SCRUMBAM has the advantage over SCRUM because is not a restrictive methodology, allowing a better designation of roles and an improved arrangement of planning meetings.

During the analysis phase of development, there were selected the topics of OOP and SE. For SE, topics selected were problem analysis, steps to solve programming problems, software processes, requirements and testing. Topics selected for OOP were algorithms, control flow and arrays.

With the selected information from the main topics, it was designed the game structure, where the steps to solve programming problems are represented each one by level in the game. The structure considered levels that are won as the player plays the game. For example, characteristics and properties of OOP, objects, actions and attributes, flow-charts and arrays, represent a level. This way, the game is divided into ten levels, each of them allowing the player to learn about the steps to solve programming problems as well as concepts of OOP.

When the structure of the serious game was designed, mockups were created to help conceptualize each level, as well as the elements contained in the serious game. The mockups were created using Balsamiq Mockups tool. At the same time, there were designed the image of the game, buttons, characters, information, etc. In Figure 2, are shown level one and level five mockups done during initial sprints.

Every game should consider some important elements, such as engagement, collaboration, participation, competition and psychological elements as motivation, ability, and trigger. With these elements in balance, the game is more attractive for the user [24].

Software KIDS uses different mechanics, dynamics and components, without losing simplicity.

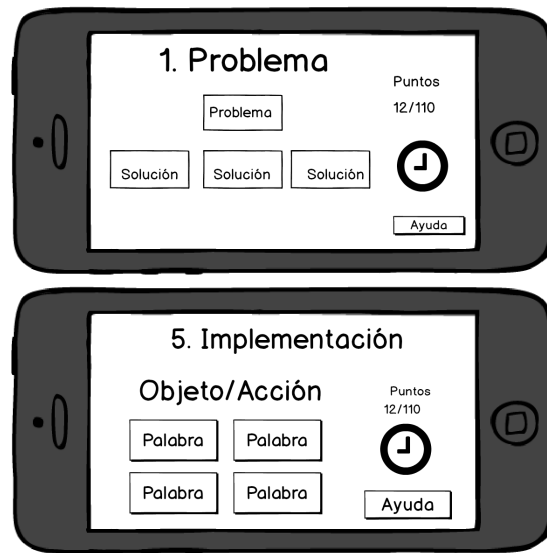


Figure 2. Sketches of levels one and five

Due to this, the serious game is mainly driven by drag-and-drop and select-true-or-false activities. This choice was chosen because they are very common in reviewed games, and allows to provide automatic and immediate feedback. These operating models are similar to puzzle games, this way the user is feeling that is playing more than learning.

One of the most important elements that were added is random triggers like bombs, stopwatch, non-stop correct or mistakes answers that provide the opportunity to introduce some changes and add or subtract points and/or time, to motivate or help player [27]. Some of the technologies and resources used to incorporate such elements were:

- UNITY: a game engine for game production.
- Adobe Illustrator: a vector drawing software.
- freepik.es: search engine of free vector designs
- soundbible.com: a search engine for free sounds.
- Balsamiq Mockup: wireframing software to make Mockups.

Software KIDS was developed in sprints. Each sprint was planned considering the elements to be included in it. Once each element was included, it was reviewed against the initial sprint planning. If the element did not satisfy initial objectives, a new plan to develop the element was created.

Development started with sprint 0. It last two days and its objective were to learn how to use UNITY. The sprints order was a team decision, and their duration was not fixed. Some of the sprints lasted one day because there due to code and elements reusability. The complete 27 sprints and their duration is shown in Table 2.

**Table 2. SOFTWAREKIDS sprint description**

Sprint ID	Sprint name	Duration (days)	Description
0	Training tools	2	Training with the main tools
1	Game analysis	3	Observation, research and analysis of several games oriented to programming and software engineering
2	Sketches creation	2	Each level sketch creation
3	Game identity creation	3	Decision making in game name and image
4	Level 2	3	Level elements and structure realization
5	Level 3	2	Level elements and structure realization
6	Level 5	2	Level elements and structure realization
7	Level 6	2	Level elements and structure realization
8	Level 4	2	Level elements and structure realization
9	Level 8	2	Level elements and structure realization
10	Level 9	2	Level elements and structure realization
11	Level 7	2	Level elements and structure realization
12	Level 10	2	Level elements and structure realization
13	Level 1	2	Level elements and structure realization
14	Level 1	1	Level elements and structure realization
15	Level 1	1	Level elements and structure realization
16	Level 4	2	Level elements and structure realization
17	Level 4	1	Level elements and structure realization
18	Level 5	1	Level elements and structure realization
19	Audio	2	Audio level choice
20	Image	1	Image level choice
21	Help section	2	Help section for all levels realization
22	Level section	2	Level access menu realization, attempt and time record realization
23	Informative sections	2	Informative section, application contact and social network realization

This table continues on the following page————>

Sprint ID	Sprint name	Duration (days)	Description
24	Integration	2	Section integration
25	Information gathering	3	Research, collection and creation of information used during games
26	Image	2	Each level image choice
27	Testing	1	Level and integration tests realization

Several sprints were oriented to the realization of graphic and audio elements to give the player immediate feedback. These elements represent an important factor in game use. A section to solve questions in each level was added because in reviewed games it is noticeable the lack of help and explanation about subjects. This lack creates frustration, desperation or boredom in players.

Other added section was a level map, presented in Figure 3, which shows how the player is progressing in the game. In the level map, the player can watch his/her best record on attempts, time, correct answers, and allowed mistakes, motivating the player's improvement on points account.

Each level is limited in time to a range between 60 and 90 seconds, depending on the played level.

To progress between levels is needed to complete all required correct answers. When the player does not achieve all correct answers on time or has more mistakes than allowed, he or she could start again. The details of each level are shown in Table 3.



Figure 3. Level map of Software KIDS

**Table 3. Description of operation of software kids**

Level ID	Level name	Operation	Time (sec)	Random trigger	Required points	Allowed mistakes
1	Problem solution	User will have to identify the adequate solution to a presented problem	60	When the user gets five non-stop correct answers, five more points are added to the player account.	20	10
2	Requirements	User will have to identify if requirement is an action or an attribute	60	There are two random triggers: a bomb which subtracts five points to player account and a stopwatch which halts time for five seconds	20	10
3	Analysis	User will have to follow clues to create selected lifecycle form to visualize it correctly	60	None	20	10
4	Design	User will have to order every activity to be done to correctly solve a problem	90	When user completes the task without mistakes, two points are added to player account	15	10
5	Objects and methods	User will have to identify among several words if the term is an object, a method or an activity	60	When user has five non-stop correct answers, two points are added to player account	20	10
6	Properties and characteristics	User will have to identify if OOP properties examples are true or false	60	When user has five correct answers, five points are added to player account	20	10
7	Control flow	User will have to select among several structures, conditional or iterative, according to a given problem and its possible solution	90	None	15	10
8	Arrangements	User will have to select the indicated cell	60	When user makes three non-stop mistakes, three points are subtracted from player account	20	5

This table continues on the following page————>

Level ID	Level name	Operation	Time (sec)	Random trigger	Required points	Allowed mistakes
9	Testing	User will have to identify if shown test is true or false	60	When user has five non-stop correct answers, time halts for five seconds	20	10
10	Maintenance and documentation	User will have to select letters and put them on correct position to complete a word	60	When user completes the word without mistakes, five points are added to player account, and when user skips a word, two points are subtracted from player account	20	10

In Figure 4, is shown the final image of these levels. Every figure is in Spanish because is the game native language.



Figure 4. Final evolution of levels one and five



Software KIDS was added to GooglePlay as a free download on the account of the Autonomous University of Zacatecas, to facilitate the installation process on mobile devices during the test and debug stage.

Test and debug stage was done with the help of ten students enrolled in the OOP course of the third semester of Computer Engineering Program, in the Autonomous University of Zacatecas. These students completed all levels and graded all of them. They also gave comments and suggestions, especially related with points, time, instructions and speed of the game, as well with some defects on level performance. In Figure 5, are shown the scores obtained in this evaluation.

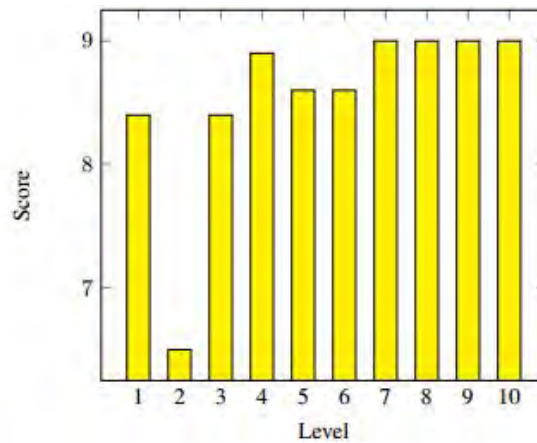


Figure 5. Student evaluation

According to Figure 5, the lowest level was level two with a grade of 6.5/10. This level was modified, as well as levels ten and four. Modifications done were related with simplicity, speed, and difficulty. Some of the comments given to game were:

1. "The games are very interesting to learn new concepts and review the basics".
2. "It is entertaining. At first, I thought, it was a very short time for each level but if they put more time the game would be too easy and I lose interest."
3. "They are good for fun and review."
4. "I think it works to describe on object-oriented programming, and with this game, is easy to understand each concept".
5. "Music is pleasant and does not exceed a pleasant volume".

## 6. Experimentation

Software KIDS was put on experimentation, with the help of Punto México Conectado, a national network of community digital training and education centers [28], and Epic Queen, a non-profit organization, dedicated to guiding, include, and educate women and girls in the technological sector [29]. Both organizations were chosen due to their importance in Zacatecas' society, and their facilities and materials.

Experimentation was done first out the Guadalupe, Zacatecas office of Punto México Conectado, with two children, and later in Epic Queen, with ten girls.

Sessions were divided into two parts: first, a small lesson about how and who created the technology, this lesson lasted one hour approximately. This hour it was discussed whom and how technology is created, children were asked about their interests, their relation with technology, and their wishes when they got older. Also, during this session importance of technology in daily life and society was shown, as well as how easy is to create it. Later, they played with Software KIDS during 50 minutes and answered a ten-minute questionnaire.

## 7. Results

Software KIDS is oriented to children older than eight years old, but during the experimentation, children of six to twelve years old also participated, as shown in Figure 5, because organizations do not restrict age in their public invitations. A mentor supported every child. Results are based on observation during experimentation and answered questionnaires, and shown that 90% of children are very interested in technology, this result is explained because they have already attended Punto México Conectado facilities and Epic Queen Code Party.

Among the aspects that were important to children when using the serious game were the sounds and the images of the application. Children qualified these aspects as positive.

A point that is important to emphasize is that children were asked which professions they would probably choose. Some of the professions mentioned were a film director, doctor, musician, teacher, veterinarian, lawyer and firefighter. Only one of the children mentioned being an engineer. The children do not know about professions related to technology, like software engineering and similar. In addition, some of the reasons for which children attend events related to technology, as the ones organized by Punto México Conectado, are:

"I like to meet more girls"

"They told me about the event and I got interested in"

"I like technology and parties"

"I want to know more about technology"

"I knew about the event and told to come"

"My parents brought me"

Children were asked to grade each level in Software KIDS in terms of difficulty, easiness, fun, and satisfaction. In Table 4 are shown votes obtained for each level.

As can be seen, level seven got the highest rates on unpleasant, tedious, entertaining, and difficult criteria. Explanation to these results is the use of English words: if, while, for, switch and case, because they do not know its translation, even when they are explained in the help section.

Level one obtained the highest score in the entertaining and easiness criteria. 100% of the children completed more than six levels, 33.3% completed all levels and 33.3% could not pass from level seven.

Table 4. Each level votes accordingly with its characteristic

	Level										None
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	
Favorite	3	1	2	1	2	2	1	2	2	3	0
Unpleasant	1	2	0	0	0	1	4	2	1	0	2
Difficult	1	1	1	1	0	0	5	2	1	0	0
Easy	8	2	2	3	3	3	0	2	2	3	0
Tedious	2	2	2	2	1	1	4	1	1	1	4
Entertaining	6	3	4	3	5	4	1	3	3	4	1

Children were asked to score Software KIDS. Results are shown in Figure 6. Average score children gave to Software KIDS was 9/10, as shown in Figure 7. Comments are given to game were:

"Games are fun"

“Level seven is difficult”

“Every level is great, except for level 7”

“Levels are difficult, but are good”

“Some levels are fun and some other are frustrating”

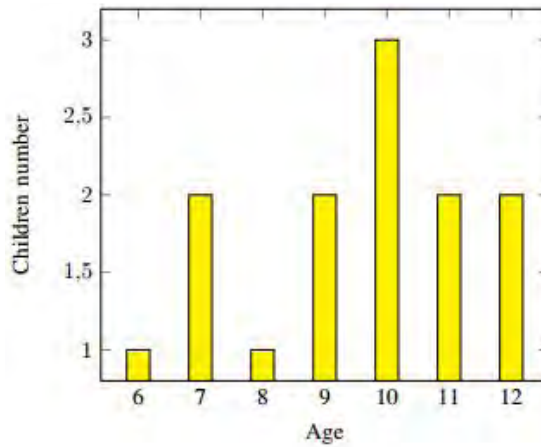


Figure 6. Children age

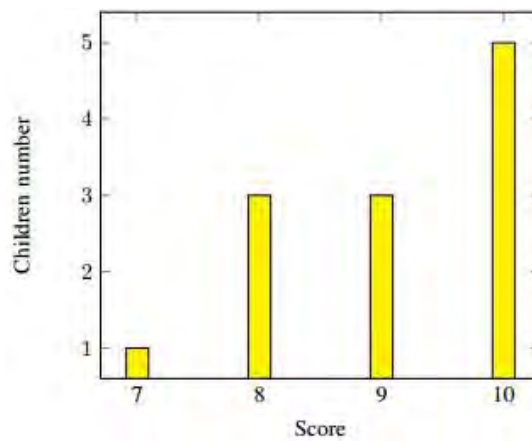


Figure 7. Software KIDS scores

“The game is boring and I prefer another type of games”

“It helps us to develop ourselves as true hackers”

“I learned many things”

“I liked it very much”

“Levels are fantastic and a little difficult”

“There is not enough time”

Mentors also observed children performance. Each mentor supported two children. Comments given by mentors are:

“High level of frustration: children began to complain about the game and required help from mentor on levels 4, 6, 7 and 8”

“The game is great and it stimulates children interest on technology”

“Level seven is difficult”

“Children did not read instructions”

“Children did not concentrate”

“Players competition”

“Lack of English language knowledge”

“Lack of interest on game and game subjects”

“Children got excited and frustrated with difficult levels”

“Time is short”

“Some activities objective was not clear”

The last version of Software KIDS was added to GooglePlay, as shown in Figure 8 on November 24th, 2015, as free download on the account of the Autonomous University of Zacatecas, to facilitate the installation process on mobile devices during test and debug stage. To this date the total downloads are 246 and 40 are active.

The users have rated Software KIDS of the scale 1 to 5, obtaining 4.9/5. The information gotten through Google Play, on June, are from the countries where Software KIDS has

been downloaded. This information stated Mexico, United States, Colombia, Bolivia, Dominican Republic, Argentina, and Ecuador as the countries with more downloads.



Figure 8. Software KIDS on GooglePlay

## 8. Conclusions and future work

The children learned about the stages of the software development, new vocabulary about SE and OOP, and the differences between objects, functions and attributes. Software KIDS motivates students to keep trying despite their mistakes. At the end, the children had a different perception of software development and SE, also they had a great and entertaining learning experience.

Based on obtained results, the game will be improved and corrected. Most important improvements are language change on levels six and seven, and game difficulty decreasing with time addition to levels seven and eight, given that 60 and 90 seconds is not enough time to accomplish correct answers required.

According to the results obtained from children and students enrolled in the OOP course, it can be concluded that in Software KIDS there are highly OOP and SE specialized terms that could be complicated for children to understand, while for students and people who know about them, the game serves to review those concepts.

Children under eight years old who interacted with Software KIDS needed the help of a mentor to solve some doubts. However, they still were able to play it without major problems. From this finding, it was realized that Software KIDS should introduce difficulty levels according to age, to avoid frustration, boredom and desertion.

In this way, promotion of SE and OOP was successful in the two sessions celebrated in Punto México Conectado and Epic Queen.

Therefore, the development of Software KIDS aided children to have fun, entertaining and a product to learn more about the SE and POO time.

As new institutions and organizations, nonprofit and governmental, in Mexico and around the world, which aim to promote programming and learning about technology development, are emerging, Software KIDS is an alternative choice to help start learning basics of programming and problem solving related to programming.

Future work also considers migration to iOS, the most used operative system in serious games related with programming teaching. Also is contemplated to have a website where children can play online to increase the number of users, mainly to use who have not a mobile device.

Another goal to achieve in Software KIDS is to create an English version because this language now represents 27.61% of the languages in serious games in Google Play, while Spanish represents only 5.25%.

The last goal to achieve is the creation of a Software KIDS version oriented to college students of the first semesters of information technology related careers.

Serious games cover a lot of areas of interest. The impact they are having is increasing and a significant growth in the coming years is assumed. Creating serious games requires a different approach than traditional game development, so new techniques and tools are required to support the process of building them. All of these bring new challenges.

## 9. References

- [1] T. Mitamura, Y. Suzuki, and T. Oohori, "Serious games for learning programming languages," Conf. Proc. - IEEE Int. Conf. Syst. Man Cybern., pp. 1812–1817, 2012.
- [2] P. Marquès, "El software educativo. J. Ferrés y P. Marqués," Comun. Educ. y Nuevas Tecnol., pp. 119–144, 1996.

- [3] I. Paliokas, C. Arapidis, and M. Mpimpitsos, "PlayLOGO 3D: A 3D interactive video game for early programming education: Let LOGO be a game," in *Proceedings - 2011 3rd International Conference on Games and Virtual Worlds for Serious Applications, VS-Games 2011*, 2011, pp. 24–31.
- [4] M. Zyda, "From Visual Simulation to Virtual Reality to Games," *Comput. IEEE Comput. Soc.*, pp. 25–32, 2005.
- [5] D. Maniega Legarda, P. Lara Navarra, and P. Yáñez Vilanova, "Uso de un videojuego inmersivo online 3d para el aprendizaje del español el caso de 'Lost in la mancha,'" *Icono 14 Rev. científica Comun. y nuevas Tecnol.*, 2011.
- [6] P. Yáñez Vilanova, "Gamificación Educativa," 2014. [Online]. Available: <https://www.youtube.com/watch?t=3697&v=vOl6HaP-uxM>. [Accessed: 03-Dec-2015].
- [7] A. Derryberry, "Serious games : online games for learning," *Serious Games*, no. 9, pp. 1–15, 2007.
- [8] G. Robles, "Una nueva herramienta ayuda a enseñar programación a niños desde los seis años," 2015. [Online]. Available: <http://www.oei.es/divulgacioncientifica/?Una-nueva-herramienta-ayuda-a>. [Accessed: 16-May-2015].
- [9] I. Sommerville and M. I. A. Galipienso, *Ingeniería del software*. 2005.
- [10] J. A. Jiménez Murillo, E. M. Jiménez Hernández, and L. N. Alvarado Zamora, *Fundamentos de Programación. Diagramas de flujo, Diagramas N-S, Pseudocódigo y Java*. Alfaomega, 2014.
- [11] K. Squire, "Video games in education," *Int. J. Intell. Games Simul.*, vol. 2, no. 1, pp. 49–62, 2003.
- [12] N. Ionel, "Critical analysis of the Scrum project management methodology," *Ann. Univ. Oradea, Econ. Sci. Ser.*, vol. 17, no. 4, p. 435, 2008.
- [13] Media Common Sense, "Common sense media," 2015. [Online]. Available: <https://www.common sense media.org/>.
- [14] A. Dantas, M. Barros, and C. Werner, "A Simulation-Based Game for Project Management Experiential Learning," *Proc. Sixt. Int. Conf. Softw. Eng. Knowl. Eng.*, pp. 19–24, 2004.
- [15] A. Jain and B. Boehm, "SimVBSE: Developing a game for value-based software engineering," in *Software Engineering Education and Training*, 2006, pp. 103–114.
- [16] E. Navarro, "An Educational, Game-Based Software Engineering Simulation Environment," 2010.
- [17] E. Navarro and A. van der Hoek, "Multi-site evaluation of SimSE," *Proc. 40th SIGCSE Tech. Symp. Comput. Sci. Educ.*, vol. 41, no. 1, p. 326, 2009.
- [18] S. Ramirez-Rosales, S. Vazquez-Reyes, J. L. Villa-Cisneros, and M. De Leon-Sigg, "A Serious Game to Promote Object Oriented Programming and Software Engineering Basic Concepts Learning," in *2016 4th International Conference in Software Engineering Research and Innovation (CONISOFT)*, 2016, pp. 97–103.



- [19] H. Partovi and A. Partovi, “‘Code Stars’ - Short Film,” 2013. [Online]. Available: <https://www.youtube.com/watch?v=dU1xS07N-FA>. [Accessed: 29-Apr-2015].
- [20] Hemendra, “5 Unbeatable Reasons Your Kid Should Be Coding,” 2014. [Online]. Available: <http://www.lifehack.org/articles/lifestyle/5-unbeatable-reasons-your-kid-should-coding.html>. [Accessed: 29-Mar-2015].
- [21] A. Rusu, R. Russell, R. Cocco, and S. DiNicolantonio, “Introducing object oriented design patterns through a puzzle-based serious computer game,” *Front. Educ. Conf.*, pp. 1–6, 2011.
- [22] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, Second Edition, vol. 2nd. 2001.
- [23] P. Arenas and J. Ricardo, “Modelo para la Motivación del Aprendizaje de la Programación utilizando Gamification,” 2014.
- [24] E. Herranz and R. Colomo-Palacios, “La Gamificación como agente de cambio en la Ingeniería del Software,” *Rev. Procesos y Métricas*, vol. 9, no. 2, pp. 30–56, 2012.
- [25] P. Letelier, M. Canós, E. Sánchez, and M. Penadés, “Metodologías Ágiles en el Desarrollo de Software,” *Val. Val. España*, pp. 1–8, 2003.
- [26] Z. Khan, “Scrumban-Adaptive Agile Development Process: Using scrumban to improve software development process,” no. May, 2014.
- [27] E. O. Navarro and A. van der Hoek, “SIMSE: An Interactive Simulation Game for Software Engineering Education,” *Cate*, 2004.
- [28] México Digital, “Punto México Conectado,” 2015. [Online]. Available: <http://pmc.gob.mx>. [Accessed: 01-Jan-2016].
- [29] Epic Queen, “Epic Queen,” 2015. [Online]. Available: <http://epicqueen.com>.

# Chapter # 25

## Representation of teaching and learning practices about embedded systems using a SEMAT kernel extension

### **Rubén Sánchez-Dams**

Grupo de Investigación  
Lenguajes  
Computacionales  
Universidad Nacional de  
Colombia  
Medellín, Colombia  
rubendams@gmail.com

### **Alexander Barón-Salazar**

Grupo de Investigación  
Galeras .Net  
Universidad de Nariño  
San Juan de Pasto,  
Colombia abaron\_98@  
udenar.edu.co

### **María Clara Gómez-Álvarez**

Grupo de Investigación  
Arkadius  
Universidad de Medellín  
Medellín, Colombia  
mcgomez@udem.edu.co

## 1. Introduction

Embedded systems are computing electronic devices with a set of limited functions to a specific use. Embedded systems are part of other products or systems such as appliances or vehicles [1]. Embedded systems contain logic determining functioning of these products. Today, embedded systems (ES) have a developing interest [2], therefore competent engineers are needed in the development and construction of such systems. Such engineers should have technical and social competencies such as collaborative work, leadership or creativity [3].

In the context of the development of embedded systems, we need to emphasize competencies [3]. This area of knowledge has traditionally been taught through two teaching strategies in higher education institutions: lectures and laboratory practices. In the first approach, professors are the main actors in the teaching-learning process and they are responsible for introducing students to the basics of topics. For the labs, professors rely on simulation projects, which apply concepts presented previously. However, there are many success stories focused on competencies of methods or practices for teaching-learning of embedded systems (TLES for its acronym) in the literature. Nevertheless, each author presents his method in a par-

ticular way, getting hard deals with different methods or its adoption. In search for TLES strategies, we find a lack of mechanisms to monitor the evolution of the competencies required by the engineer in the industrial environment. There is a lack of a common framework to facilitate the teaching-learning process and the achievement of learning goals.

Authors present a representation mechanism for processes of TLES with the purpose of monitoring the development of competencies, initially described in [4]. The mechanism is based on the SEMAT kernel as a universal framework for representing software engineering practices, defining an extension to such kernel [5]. In [4] we were focused on a methodology to extend the SEMAT kernel toward other disciplinary domains. In this chapter we focus on exemplify the use of the representation mechanism proposed for TLES processes. In addition, we propose specific validation criteria of the proposal using a set TLSE methods found in the literature.

## 2. Theoretical Framework

### 2.1 Concept of Practice

In software engineering context, practice is a common concept associated with different structures and definitions [6, 7]. Capability Maturity Model Integration for Development (CMMI-DEV) defines a set of process areas; a group of practices forms each area. Implementation of such practices satisfies a set of generic or specific goals [8]. In the case of Eclipse Process Framework Composer (EPFC), a practice is a documented approach to solving a set of common problems. This practice definition is related to i) work products as a result of practice application, ii) tasks confirming the steps of the practice development, iii) guidelines for practice implementation, and iv) roles assumed by participants of a practice [9].

The OMG (Object Management Group) standard, Essence – Kernel and Language for Software Engineering Methods–known as SEMAT kernel–, defines a software practice as a repeatable approach with a specific goal [5]. A software practice is a practitioners’ guideline about what should be done for achieving this goal, expresses in terms of expected results. Besides, such practice seeks to ensure the goal understanding and the verification of obtained results by the work team. The practice elements proposed by this standard are presented in the Figure 1.

Finally, Rational Unified Process (RUP) presents a practice like a concept with two dimensions: horizontal –for dynamic aspects–, and vertical –for static aspects–. In





Figure 2. Teaching and Learning practice components

### Teaching Strategies

Set of resources, techniques and procedures used by the professors for generating meaningful learning in their students [13]. A systematic process of planning, design, implementation and evaluation of teaching and learning activities is one of the ways for archiving such meaningful learning. Additionally, the design of teaching strategies should take into account promotes in the students the abilities of observation, analysis, synthesis, hypothesis formulation and problem solving. In other words, the main goal of teaching strategies is motivated the students to discover knowledge by themselves.

In this way, a student becomes the main actor of their learning process and the professor is just a guide or facilitator of this process.

### Learning Goals

Learning goals refer to the cognitive activities that students should be able to do at the end of the application of teaching strategies. Such goals are related to the knowledge, abilities and attitudes expected by the professor in the curriculum planning and previous courses [14].

### Content

Content is the set of subjects characterizing a phenomenon or event. In the context of a course, content is the set of specific subjects described in the syllabus and correspond

to the knowledge to share with the professor with their students for developing technical competencies.

### **Support Platform**

Support platform includes a set of technological tools for complement teaching-learning process. Such tools allow students learn and work by themselves, using the resources and activities available in these applications [15, 16].

### **Learning environment**

Learning environment refers to the different contexts, physical locations and conditions in which students learn. According to Forneiro, a learning environment comprises four dimensions [17]:

- a. Physical dimension: the physical space, structural conditions and objects in the space where the learning process take place.
- b. Functional Dimension: this dimension refers to the use of the space and the type of activities carried out.
- c. Temporal dimension: time assigned to the different learning activities.
- d. Relational Dimension: such dimension is associated with the set of relationships established in the classroom between students-students and professor-students.

### **Assessment**

Assessment is a key component of the teaching-learning process [18]. Assessment purpose is verifying if the students develop the competencies expected after the application of the teaching practices. Such activity is traditionally carried out via written exams, but other evaluation methods are appearing like group activities or ludic workshops [19, 20].

## **2.3 Embedded Systems**

Products related to automotive, industrial automation, network equipment, mobile devices, and appliances, incorporate embedded systems [5, 6]. An embedded system integrates hardware and software on a computer with a particular purpose [21]. The software

has the feature to directly access hardware resources. The Software determines the logical behavior of the embedded system. However, logical functionality can be implemented directly in hardware according to performance needs [22].

To design an ES, engineers take into account restrictions such as processing capacity, ports, real time, or power consumption. Usually, the system architecture is limited according to the proposed use of an embedded system; the system is built strictly with the functionality of its purpose. Engineers use different approaches and co-design methods to develop embedded systems [21] incorporating different levels of abstraction: a) software components level, b) hardware blocks level, c) software/hardware integration level, and d) system level [1].

## 2.4 Essence Standard like a Theoretical Framework to Practices Representation

SEMAT initiative provides a kernel and a language for creation, use and improvement of software engineering methods. The kernel and the language are defined in the Essence standard [5]. Such standard allows describing foundations of methods and practices of software engineering, so they can be compared, evaluated, used, adapted, simulated, measured and investigated by professionals, academics and researchers.

Essence standard defines three areas of concern: customer, solution and endeavor. The areas “customer” and “solution” are related to problem and solution domain, respectively. Meanwhile, the resources involving in the solution development are part of the area “endeavor”. The kernel has these main constructs associated to a) things that work with –alphas–, b) what are done–activity spaces– and c) capacity to perform the work–competencies–. These constructs are discriminated according to these areas of concern.

Essence defines seven alphas with which progress and health of the most important elements of software engineering are measured. It is possible to define new sub-alphas to describe particular situations. In Figure 2, these alphas and their relationship are shown.

Activity spaces are representations of essential things to do. Such representations provide descriptions of challenges that a team faces for the development, maintenance and support of software systems. They also provide a description of things that the team will take into account to meet these challenges. Examples of activities spaces are: explore possibilities, understand requirements, and support the team for the areas of concern customer, solution, and endeavor, respectively. Competencies are representations of the key skills required by software engineers. As an example, in the area of concern endea-

vor, the team should organize the work, because of the competencies Leadership and Management are needed.

In relation to theory construction, a theory is a common conceptual framework to structure and organize facts and knowledge in a concise and accurate manner [23]. Theories are useful for facilitating the understanding of several phenomena and the knowledge exchange among researchers and professionals. Theories main components are: i) constructs–essential conceptualization of a specific domain–, and ii) propositions–structural and dynamic relationship between constructs means by equations, statements or laws– [23, 24].

Essence standard constitutes a framework for representing software engineering practices and methods. For this reason, Essence is defined as a theoretical approach of software engineering methods, identifying constructs and propositions for any method or practice [25]. Scalability, extensibility, and ease-of-use features of the kernel allows being applied for

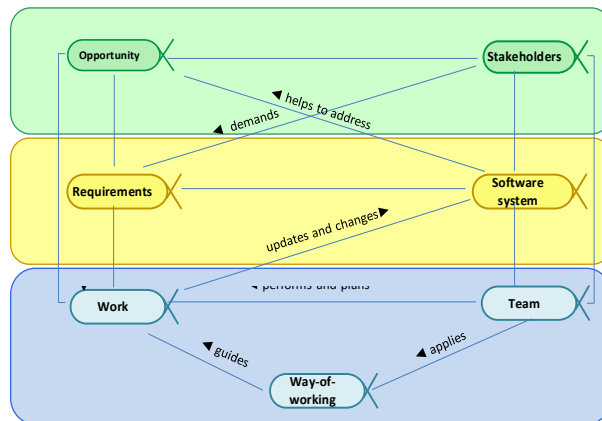


Figure 3. Alphas and their relationships representing methods and practices of other disciplines

### 3. Previous Work

In this section, the authors present some approaches about embedded systems teaching and learning practices. Such approaches include activities like Arduino practices and reality simulation of embedded system development industry, besides of lectures and laboratory practices.

Chenard et al. describe a teaching methodology for embedded systems supported by project-based learning [3]. Such methodology and the associated laboratory meets the



needs of learning wireless embedded systems. Additionally, they present a hardware platform like a support for practice course projects of students. Mitsui et al. propose a student experiment method for implement embedded systems [1]. This method helps students to develop design competencies like system modeling and module design.

In relation to embedded systems course design, Nooshabadi and Garside present a course design proposal seeking a learning environment that simulates the reality of embedded systems development industry [26]. This proposal includes the integration of processors, dedicated coprocessors and software for creating solutions for electronic devices like smartphones and tablets. Meanwhile, Bareno describes a three-course program for embedded systems teaching [27]. This program is based on software platforms for embedded system development and hardware open source looking the promotion of student competencies related to conception, design and implementation of this type of systems.

Similarly, Jamieson proposes an embedded systems course founded on project-based learning and Arduino as an implementation platform [28]. Such platform exposes students to complex problems and challenges for embedded systems development. Finally, Rover et al. present a set of course of embedded systems designed using a learning model based on cognition levels of Bloom's taxonomy. This proposal includes the incorporation of elements like learning labs, learned lessons repository and new technologies [29].

The revision of previous work includes the construction of a pre-conceptual schema for each proposal of embedded systems teaching. The goal is of this analysis is identifying the main concepts (constructs) and relationships (propositions) of these TLPES. For example, the pre-conceptual schema for the proposal of Nooshabadi and Garside is shown in Figure 4 [26].

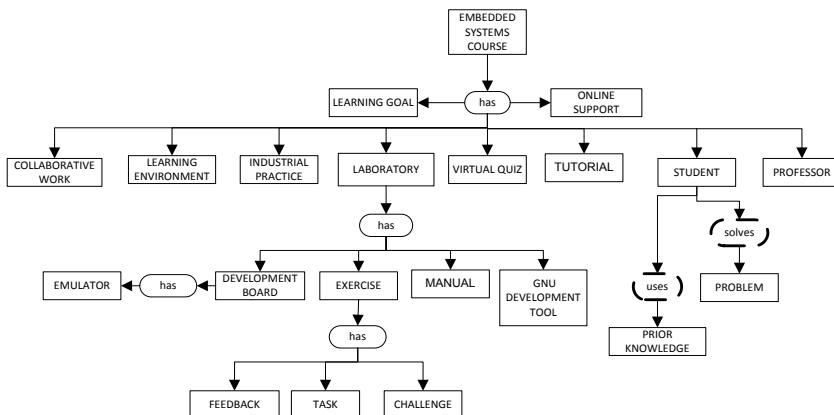


Figure 4. Pre-conceptual schema summarizing Nooshabadi and Garside teaching and learning proposal for embedded systems [26].

# 4. An Extension of the Semat Kernel for Representing Teaching and Learning Practices about Embedded Systems

## 4.1 Methodology

Methodology for extending the Semat kernel consists of two approaches. The first of them is based on theory construction procedures proposed by Sjøberg, Dyba, Anda and Hannay [20], and Ekstedt [21]. In the second approach, the authors use the tools of traditional scientific methodology to implement the procedures proposed in the first approach. Such approaches were organized in three phases: definition, construction, and validation. Here, we present the table I where we summarize the relations of the two approaches and phases.

Table 1. Methodological approaches to formulate the Semat extension

Phases	Sjøberg steps approach [23]]	Traditional scientific approaches
Definition	Define the constructs of the theory	Systematic literature review [30]
	Define the propositions of the theory	
Construction	Provide an explanation to justify the theory	- Analyzing papers and their representation in pre-conceptual schemas [31]. - Terminology approval of the concepts identified in the pre-conceptual schemas
	Determine the scope of the theory	Defining criteria for the extension of the SEMAT kernel
Validation	Test the theory by empirical studies	- Empirical application of the extension model proposed - Using executable pre-conceptual schemas for representing and instantiating the main constructs of a theory [25]

Authors based the methodology on two hypotheses: a) there is a set of essential, common constructs and propositions to all TLPEs; and b) such constructs and propositions can be discovered in a finite subset of items available in the literature. Such hypotheses are proved to apply the methodology.

### 4.1.1 Definition Phase

The systematic literature review was performed using the guidelines shown in [30]. In this vein, the research question was asked: How embedded systems development are taught? The query string in Google Scholar was ((teach or teaching) AND “embedded system”), and 5,140 results were found. For analysis, a format for data extraction was proposed; finally, the main concepts and relationships of the selected documents were synthesized.

### 4.1.2 Construction Phase

From the information extracted from the documents, the authors express the controlled speech by using pre-conceptual schemas. With this technique, ambiguities, concepts and relationships are clarified. Then, terminological homologation to extract common constructs and propositions to analyzed documents is performed. Then, the scope of the extension for representing TLES is defined, based on the SEMAT kernel. Scoping is done with the classification of constructs–concepts–and propositions–relations–extracted from the terminological homologation, which meets the following criteria: a) Extend the SEMAT Kernel; b) Orthogonality of Constructs; c) Generality; and d) Establishment of States, Checklists and Descriptions of Concepts by analogies to the SEMAT kernel.

### 4.1.3 Validation Phase

The first chosen approach is pragmatic validation through successfully representing different TLES not used to generate the extension. Validation is left raised and involves the use of the kernel and the extension to represent a set of TLES. Examples of representations of TLPES practices do not include in the systematic literature review are shown in Section V of this chapter.

The second approach consists of using executable pre-conceptual schemas for representing the SEMAT kernel [25]. Figure 5 shows a pre-conceptual schema for Semat initiative understood as a general theory about software engineering. Then, we can define a set of constructs and propositions about Semat extension for representing TLES according to such schema. In table II, we show five propositions related to the alphas Embedded system and Learning Environment. Readers can see these relationships in Figure 6. Validation is to ensure that the propositions are present in a set of TLES not used to generate the extension.

To deepen in the methodology to extend the SEMAT kernel toward other disciplinary domains, we recommend readers to see [4].

## 4.2 SEMAT extension definition for teaching and learning practices about embedded systems

For this extension, an exception to the use of the kernel is raised: the object of study is set; the Software System alpha is established as a construct within a broader alpha called Embedded System, which it is related with another construct called Hardware System. Embedded System Alpha replaces the Software System in terms of its relationships with the other alphas. In addition, a new alpha called Learning Environment is proposed, taking into account its importance in the teaching and learning process—criteria of orthogonality of constructs of the construction phase; such environment includes the space, the rules of use of this space, and the relationships established in the classroom. The learning environment aims to study the embedded system and defines the work of teaching and learning, which is executed by the team representing the professor and students. In turn, the team participates in the learning environment. In addition, authors identified states, verified alphas, and proposed a work product to the learning environment. Figure 6 shows the relationships among alphas in the extension. Figure 7 shows the alphas and their sub-alphas. Figure 8 shows the card for the Learning environment alpha. Table III shows the states of the Embedded System alpha. Finally, table IV shows a consolidated of the alpha and sub-alphas.

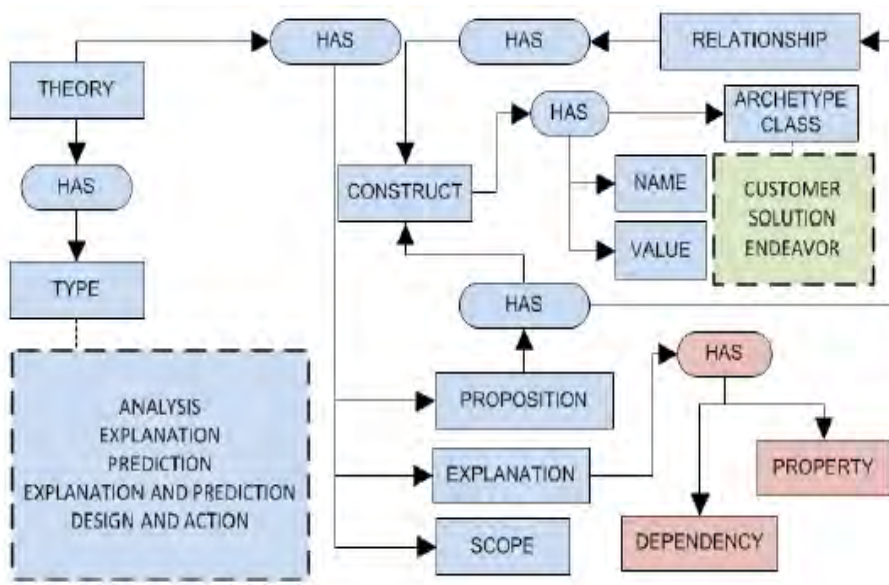


Figure 5. The theoretical pre-conceptual schema for Semat initiative [25].

Table 2. Some constructs and propositions of SEMAT extension for representing TLES

THEORY			
PROPOSITION			
CONSTRUCT			RELATIONSHIP
Name	Value	Arche-type Class	Name
Alpha	Team	Endeavor	participates
Alpha	Learning Environment	Solution	
Alpha	Learning Environment	Solution	delimits
Alpha	Work	Endeavor	
Alpha	Embedded system	Solution	object of study
Alpha	Learning Environment	Solution	
Alpha	Embedded system	Solution	fulfills
Alpha	Requirements	Solution	
Alpha	Work	Endeavor	changes
Alpha	Embedded system	Solution	

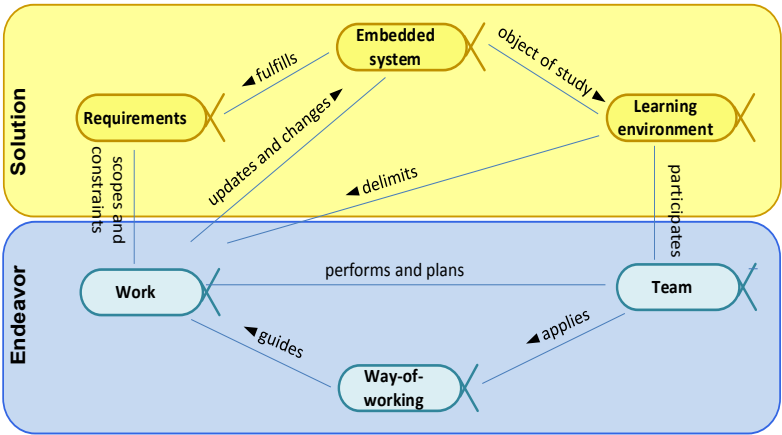


Figure 6. Relations among alphas with the proposed extension

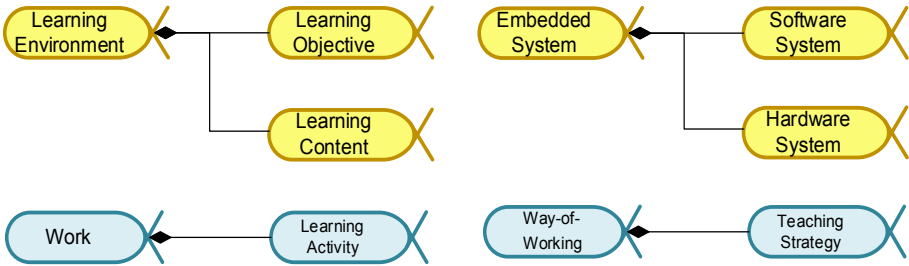


Figure 7. Sub-alphas of the proposed extension

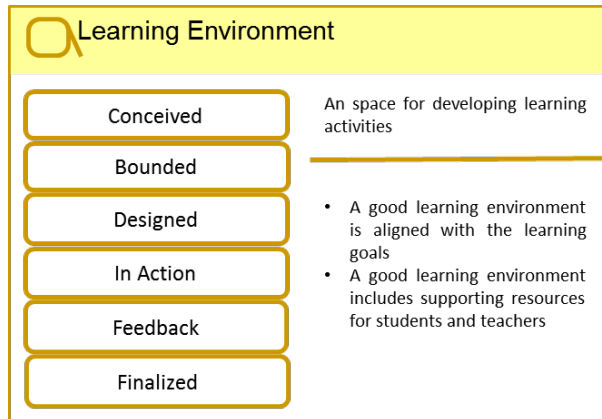


Figure 8. Learning environment alpha card

Table 3. States of learning environment alpha

State	Description
Conceived	The agreed need for a learning environment
Bounded	The purpose and contents of the learning environment are clear
Designed	The learning environment is fully described by modeling mechanisms
In Action	The learning environment is usable and allows evaluation
Feedback	The actors in the teaching-learning process identifies opportunities to improve the learning environment
Finalized	The learning environment meets the agreed requirements

Table 4. Alphas proposed

Alpha name	Sub-alpha	Description	States
Learning environment		An space for developing learning activities	<ul style="list-style-type: none"> <li>- Conceived</li> <li>- Bounded</li> <li>- Designed</li> <li>- In Action</li> <li>- Feedback</li> <li>- Finalized</li> </ul>
Learning Goal	x	Learning objectives correspond to what the student should be able to demonstrate at the end of a process of teaching and learning	<ul style="list-style-type: none"> <li>- Defined</li> <li>- Presented</li> <li>- Addressed</li> <li>- Assessed</li> <li>- Completed</li> </ul>
Learning Content	x	Set of concepts that are addressed in a training process	<ul style="list-style-type: none"> <li>- Defined</li> <li>- Bounded</li> <li>- Ready</li> <li>- Presented</li> <li>- Assessed</li> </ul>

This table continues on the following page →

Alpha name	Sub-alpha	Description	States
Hardware System	x	Electronic system for the generation, transmission, handling, processing or storage of analog or digital signals	<ul style="list-style-type: none"> <li>- Architecture selected</li> <li>- Demonstrable</li> <li>- Usable</li> <li>- Made and integrated</li> <li>- Tuned</li> <li>- Retired</li> </ul>
Learning Activity	x	Actions or tasks performed by the student to learn content and develop competencies	<ul style="list-style-type: none"> <li>- Conceived</li> <li>- Designed</li> <li>- Made</li> <li>- Completed</li> <li>- Assessed</li> <li>- feedback</li> </ul>
Teaching Strategy	x	Procedures or resources used by professors to promote meaningful learning	<ul style="list-style-type: none"> <li>- Conceived</li> <li>- Bounded</li> <li>- Planned / Designed</li> <li>- Applied</li> <li>- Assessed</li> <li>- Feedback</li> </ul>

We propose a new space activity named “do learning activity” in the area of interest endeavor. The other activity spaces of SEMAT kernel meet the needs of TLES. In the Figure 9, it shows the new activity space.

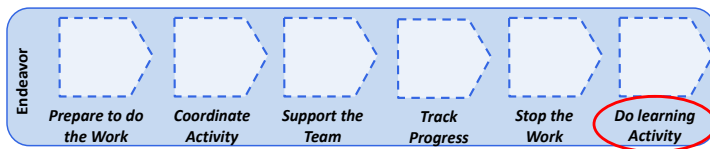


Figure 9. Activity spaces “Do learning activity”

## 5. Representation of methods or practices for teaching/learning of embedded systems

In this section, we present two methods of TLES. These methods are described in papers that were found in the literature with the same search string presented in the item “Definition Phase” of the methodology. The papers represented with extension are different from those used for generating the representation mechanism proposed in this chapter. Here we present a representation of each method using the proposed extension of the SEMAT kernel for representing teaching and learning practices about embedded systems.

## 5.1 The Use of Video-Game Devices as a Motivation for Learning Embedded Systems Programming (VG-TES)

VG-TES is an approach in use since 2004 at the University of Granada, Spain, where portable video game consoles are used as the platform to teach embedded systems programming. Besides being based on embedded processors and custom hardware (video processors, media codecs, etc.) and incorporating more devices than the average development boards at a lower price, the most important feature of these consumer electronics is the attraction that many students feel with them, having spent pleasant times on them with relatives and friends. Another advantage of using a video game console as the development platform is that, because of the high sales volume of the videogame market, many students will probably have already purchased or would like to purchase one for recreational purposes, or have easy access to one via a relative or friend. In this work, the representation of a method for teaching and learning embedded systems described VG-TES using the Semat kernel extension is explored.

### 1. Practice VG-TES

The VG-TES method proposes conducting “laboratory practices” on a course of embedded systems to obtain as a final product an embedded system in the form of video game. VG-TES method like a Semat practice is represented in Figure 10.

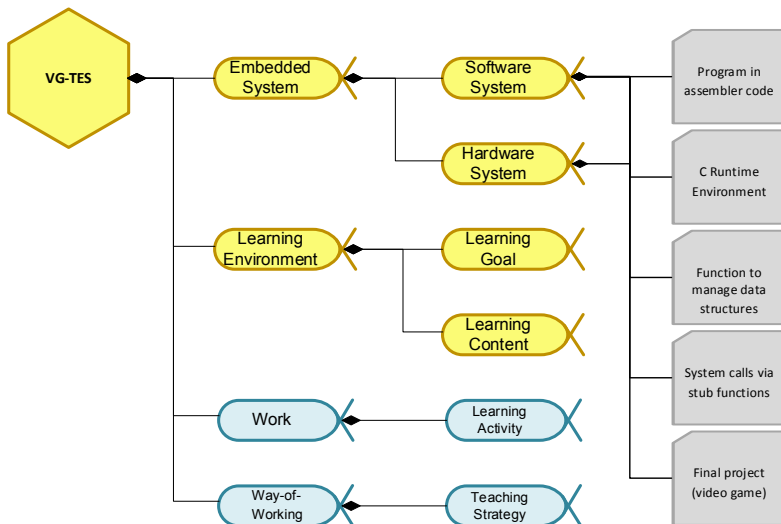


Figure 10. Graphical syntax of the VG-TES practice



## 2. Monitoring practice by alphas

Each of the sub-alphas and alphas that are part of the extension are instantiated in elements of the VG-TES proposal. This is a way of demonstrating that the elements that are part of a practice of teaching and learning embedded systems can be represented by elements that are part of the kernel and the Semat extension. Thus, it is possible to exploit the advantages of the Semat kernel to monitor the health and progress of the construct of the embedded system and the teaching process. Table V shows the elements, that instantiate the sub-alphas which is part of the extension, are identified.

## 3. Practice activities

The activities proposed by VG-TES integrate activity spaces that are part of the Semat kernel. This integration allows defining, monitoring and evaluating the process proposed by VG-TES as practice for teaching embedded systems. Relations between VG-TES practice, activity spaces, activities and work products are shown in Figure 11.

**Table 5. Alphas, sub-alphas and instances in VG-TES**

Alpha	Sub-alpha	Instance in VG-TES
Learning Environment	Learning Goal	<ul style="list-style-type: none"><li>- Identify the features that distinguish embedded systems from general-purpose computing systems</li><li>- Select, configure, and use embedded systems debugging and development tools</li><li>- Develop firmware for simple embedded systems</li><li>- Develop peripheral drivers at different levels of abstraction</li><li>- Optimize embedded software to maximize its performance and minimize its power consumption</li></ul>
	Learning Content	<ul style="list-style-type: none"><li>- Features of the embedded systems</li><li>- Embedded software</li><li>- Embedded systems debugging</li><li>- Embedded systems development tools</li><li>- Firmware for embedded systems</li><li>- Peripheral drivers</li></ul>
Embedded System	Software System	<ul style="list-style-type: none"><li>- Video Game</li></ul>
	Hardware System	<ul style="list-style-type: none"><li>- Video Game Console</li></ul>
Work	Learning activity	<ul style="list-style-type: none"><li>- Lecture session</li><li>- Laboratory session</li><li>- Laboratory assignments</li><li>- Homework</li></ul>
Way of working	Teaching Strategy	<ul style="list-style-type: none"><li>- Introduction to Platform and Toolset</li><li>- Development of a Basic C Runtime Environment</li><li>- Device Handling</li><li>- System Calls</li><li>- Final Project</li></ul>

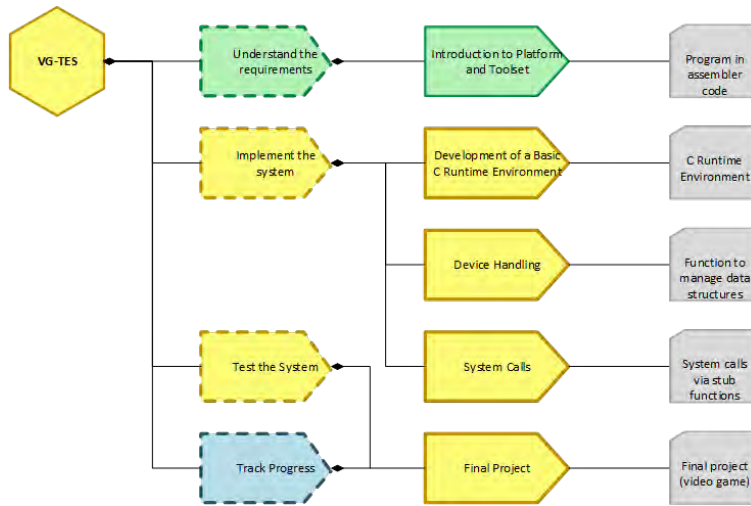


Figure 11. VG-TES practices, Activity Spaces, Activities, and work product.

## 5.2 Definition of the TLPES Structure Learning of Embedded System Design, Simulation and Implementation: A Technical Approach

“Learning embedded systems based on simulation” is a teaching-learning method of embedded systems based on the experience of the authors in the Federal University of Technology in Nigeria. Such method of teaching and learning is based on the use of Computer Aided Design (CAD) tools for circuit simulation. The method is aimed at embedded systems introductory courses. Authors propose the use of a virtual microcontroller-based circuit experiments method. They use proprietary brands in microcontroller and the CAD tool. In the method, authors use a paradigm of teaching/learning of “learn-while-doing”.

### 1. Method of learning embedded systems based on simulation

In the methodology, proposed activities jointly address the software/hardware development, due to the low complexity of the projects presented in the original article. But for clarity, we separate the activities of software and hardware in the representation made with the kernel Semat and the proposed extension. In the article, authors do not mention learning contents or objectives of a course of embedded systems where the method is applied. Accordingly, the representation of the methodology lacks the sub-alphas “Learning Goal” and “Learning Content”. “Learning embedded systems based on simulation” method was divided in two practices “Maintaining course” and “ES building

based on simulation”. Then in Figure 12, we represent the corresponding alphas, sub-alphas and work products associated with the practices.

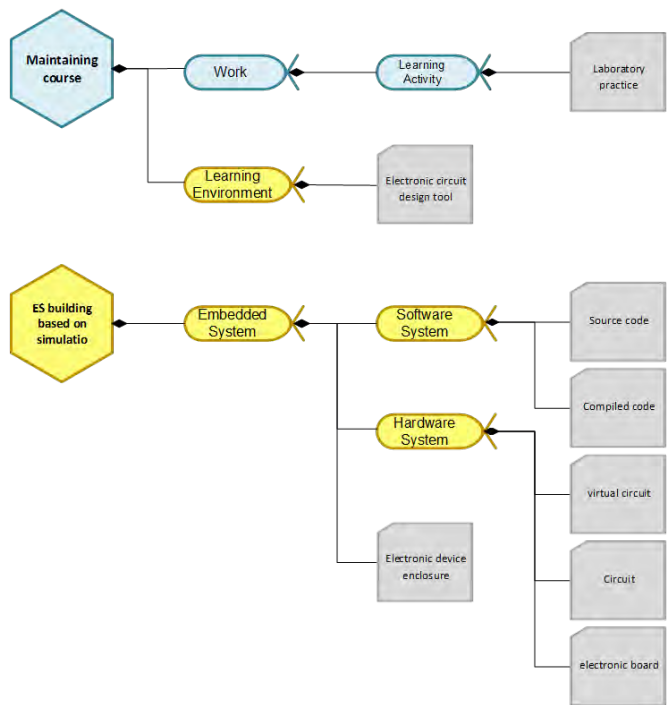


Figure 12. Graphical syntax of the “Learning embedded systems based on simulation” method

## 2. Monitoring practice by alphas

Table 6 shows the elements, that instantiate the sub-alphas which is part of the extension, are identified. Instances are used for tracking laboratory practices.

Table 6. Alphas, sub-alphas and instances in “learning embedded systems based on simulation” method

Alpha	Sub-alpha	Instance in VG-TES
Embedded System	Software System	- Source code of Learning activities
	Hardware System	- Circuits of Learning activities
		- Electronic device enclosure of Learning activities
Work	Learning activity	<ul style="list-style-type: none"> <li>- LED chasing and 3x3x3 LED Cube</li> <li>- LCD module displays message</li> <li>- Microcontroller-based digital thermometer</li> <li>- 7-segment display countdown timer</li> <li>- Analog-digital converter (ADC) module</li> </ul>

### 3. Practice activities

The development process of embedded system is divided into four stages: a) circuit design and simulation, b) system demonstration circuit, c) implementation Circuit, and d) packaging. Relations between practices, activity spaces, activities and work products are shown in Figure 13.



Figure 13. Activity Spaces, activities, and work products of the “Learning embedded systems based on simulation” method

## 6. Conclusions

In this chapter is presented, for the discussion of the academic community, a methodology to extend the SEMAT kernel. The Essence standard could be used as a framework to represent the teaching and learning practices about embedded systems. A prototype of the kernel extension was achieved from a limited set of bibliographic sources of the universe of teaching practices available for embedded systems. In the exercise, it was identified that the constructs and propositions of the SEMAT kernel are sufficiently general, allowing the representation of the new domain with the addition of a few extensions. In addition, new constructs and propositions of the proposed extension are in the rest of the twenty-eighth papers analyzed. In addition, the authors propose to study the structure of SEMAT kernel as a general framework for the representation of practices and methods from other discipline domains than the software engineering domain, like teaching and learning practices about embedded systems. The results demonstrate the scalability, extensibility and ease-of-use of Essence for representation, usage and improvement of practices and methods of other disciplines.

The SEMAT kernel extension presented for representing TLES allowed to verify the two hypotheses related to the existence of a common set of constructs and propositions to different TLES, extracted from scientific papers that describe teaching and learning practices about embedded systems. However, this verification is biased because it is necessary to broaden the base or universe of documents reviewed, to verify empirically the potential of this extension to represent any TLPES.

As lines of future work, we propose:

- Increase the scope of the systematic review of the literature on teaching and learning practices about embedded systems based on increasing the number papers to review from recognized databases like as IEEE or ACM and using other search strings that allow refining results.
- Analyze the “competency” construct, which is part of the domain of teaching-learning process, and it refers to the competencies to promote students as well as the competencies required to lead the teaching process. In the first case, you can define indicators to verify the achievement of competency by monitoring the level of achievement of the abilities associated. Another alternative is to measure the progress of the teaching-learning process through the states of other subalphas as Learning Objective.

## 7. References

- [1] H. Mitsui, H. Kambe, and H. Koizumi, "Use of Student Experiments for Teaching Embedded Software Development Including HW/SW Co-Design," *IEEE Transactions on Education*, vol. 52, no. 3, pp. 436–443, Aug. 2009.
- [2] S. Ordóñez, "Empresas y cadenas de valor en la industria electrónica en México," *Economía UNAM*, vol. 2, no. 5, pp. 90–111, 2005.
- [3] J.-S. Chenard, Z. Zilic, and M. Prokic, "A Laboratory Setup and Teaching Methodology for Wireless and Mobile Embedded Systems," *IEEE Transactions on Education*, vol. 51, no. 3, pp. 378–384, Aug. 2008.
- [4] R. Sánchez-Dams, A. Barón-Salazar, and M. C. Gómez-Álvarez, "An Extension of the SEMAT Kernel for Representing Teaching and Learning Practices about Embedded Systems," in *2016 4th International Conference in Software Engineering Research and Innovation (CONISOFT)*, 2016, pp. 39–46.
- [5] OMG Group, "Kernel and Language for Software Engineering Methods (Essence)." 02-Nov-2014.
- [6] J. G. Guzmán, D. Martín, J. Urbano, and A. de Amescua, "Practical experiences in modelling software engineering practices: The project patterns approach," *Software Qual J*, vol. 21, no. 2, pp. 325–354, Jul. 2012.
- [7] C. Passos, D. S. Cruzes, T. Dybala, and M. Mendonça, "Challenges of Applying Ethnography to Study Software Practices," in *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, New York, NY, USA, 2012, pp. 9–18.
- [8] CMMI Product Team, "CMMI for Development, Version 1.3," *Software Engineering Process Management Program*, Nov. 2010.
- [9] The Eclipse Foundation. (2014). EPF Practices, Eclipse Process Framework Composer. [Online]. Available: <http://epf.eclipse.org/wikis/epfpractices/index.htm>.
- [10] Rational Unified Process: Best practices for software development teams. [Online]. Available: [https://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251\\_bestpractices\\_TP026B.pdf](https://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf)
- [11] P. Grossman, C. Compton, D. Igra, M. Ronfeldt, E. Shahan, and P. Williamson, "Teaching practice: A cross-professional perspective," *Teachers College Record*, vol. 111, no. 9, pp. 2055–2100, 2009.
- [12] C. Zapata and I. Jacobson, "A first course in software engineering methods and theory," *Dyna*, vol. 81, no. 183, pp. 231–241, 2014.
- [13] P. E. G. Férrez, "Un acercamiento al trabajo colaborativo," *Revista Iberoamericana de Educación*, vol. 35, no. 2, 2005.
- [14] R. M. Harden, "Learning outcomes and instructional objectives: is there a difference?" *Medical teacher*, vol. 24, no. 2, pp. 151–155, 2002.

- [15] Garcia, A., M. Daneri, Investigación e innovación en el conocimiento educativo actual, En R. I. Roig y J.E. Blasco (Coord.),2008.
- [16] H. Chen and K. Damevski, "A teaching model for development of sensor-driven mobile applications," in Proceedings of the 2014 conference on Innovation & technology in computer science education, 2014, pp. 147–152.
- [17] M. L. I. Forneiro, "Observación y evaluación del ambiente de aprendizaje en educación infantil: dimensiones y variables a considerar," Revista Iberoamericana de educación, no. 47, pp. 49–70, 2008.
- [18] A. M. Vivar Quintana, A. B. González Rogado, A. B. Ramos Gavilán, I. R. Martín, M. Ascensión, R. Esteban, T. A. Zorrila, and J. F. Martín Izard, "Application of rubric in learning assessment: a proposal of application for engineering students," in Proceedings of the First International Conference on Technological Ecosystem for Enhancing Multiculturality, 2013, pp. 441–446.
- [19] M. Spellings, A test of leadership: Charting the future of US higher education. Washington DC: US Department of Education, 2006.
- [20] D. Golden, "Colleges, accreditors seek better ways to measure learning," Wall Street Journal, pp. 738027–298, 2006.
- [21] P. Marwedel, Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems, Edición: 2. Dordrecht: Springer, 2010.
- [22] J. Dimitrov, "Developing semantics of Verilog HDL in formal compositional design of mixed hardware/software system," PhD thesis, Software Technology Research Laboratory, De Montfort University, 2002.
- [23] D. I. K. Sjøberg, T. Dybå, B. C. Anda, and J. E. Hannay, "Building theories in software engineering," in Guide to advanced empirical software engineering, F. Shull, J. Singer, and D. I. K. Sjøberg, Eds. London: Springer, 2008, pp. 312–336.
- [24] M. Ekstedt, "An empirical approach to a general theory of software (engineering)," in 2013 2nd SEMAT Workshop on a General Theory of Software Engineering (GTSE), 2013, pp. 23–26.
- [25] C. M. Zapata-Jaramillo, "An executable pre-conceptual schema for a software engineering general theory," in Software Engineering: Methods, Modeling, vol. 3, C. M. Zapata-Jaramillo and L. F. Castro-Rojas, Eds. Centro Editorial de la Facultad de Minas, 2014, pp. 3–7.
- [26] S. Nooshabadi and J. Garside, "Modernization of teaching in embedded systems design-an international collaborative project," IEEE Transactions on Education, vol. 49, no. 2, pp. 254–262, May 2006.
- [27] C. I. C. Bareno, "Teching/Learning Methods for Embedded Systems Using Copyleft Hardware," IEEE Latin America Transactions, vol. 9, no. 4, pp. 503–509, 2011.
- [28] P. Jamieson, "Arduino for teaching embedded systems. are computer scientists and engineering educators missing the boat?" Proc. FECS, pp. 289–294, 2010.

- [29] D. T. Rover, R. A. Mercado, Z. Zhang, M. C. Shelley, and D. S. Helvick, "Reflections on teaching and learning in an advanced undergraduate course in embedded systems," *IEEE Transactions on Education*, vol. 51, no. 3, pp. 400–412, 2008.
- [30] B. Kitchenham, "Procedures for performing systematic reviews," Keele, UK, Keele University, vol. 33, no. 2004, pp. 1–26, 2004.
- [31] C. M. Zapata-Jaramillo, A. Gelbukh, and F. Arango-Isaza, "Pre-conceptual schema: A conceptual-graph-like knowledge representation for requirements elicitation," in *MICA 2006: Advances in Artificial Intelligence*, Springer, 2006, pp. 27–37.



Este libro se terminó de diseñar  
el 22 de mayo de 2017  
en la Unidad de Comunicaciones y Protocolo de la  
Universidad de San Buenaventura, sede Bogotá



UNIVERSIDAD DE  
SAN BUENAVENTURA  
SEDE BOGOTÁ

**EB**  
EDITORIAL  
BONAVENTURIANA

Diseño e impresión  
Unidad de Comunicaciones y Protocolo de la Universidad de San Buenaventura, sede Bogotá