PhD Dissertation

International Doctorate School in Information and Communication Technologies

DISI - University of Trento

Tape Mbo'e: A Service Oriented Method

Ilse María Grau Yegros

Tutors
Prof. Luca Cernuzzi
Prof. Adolfo Villafiorita

April 2015

Abstract

In developing countries, normally, different e-government applications do not exchange data each other, affecting the quality of service provided to citizens and the transparency. This situation has motived us to focus on the development of applications in such domain and, more specifically, on the interoperability among these applications. The interoperability has been implemented in e-government context of multiple countries through the Service-Oriented Computing (SOC). In addition to the interoperability, SOC provides considerable benefits. However, developing a service-based e-government application (SBeA) is a complex challenge and requires a software engineering method to manage its development process. The potential method should take into account not only design, construction, and maintenance of SBeA, but also it should consider the context in which it will be used (i.e, developing countries). In fact, in these countries, the scarcity of economic resources and qualified professionals can impose constraints to carry out e-government projects. Thus, the adopted methods have to maintain the costs low and to consider aspects related to long-term sustainability of the applications from the beginning. These issues suggest the adoption of Agile Methods (AMs) that have been proved to offer benefits in different projects in developing countries. However, they do not include SOC characteristics. Therefore, we have proposed Tape Mbo'e (TME)[1], that is an extension of the agile method "OpenUP" to support the construction and the maintenance of SBeA in developing countries. TME has been used in five case studies, which embraced both academia and public organizations in Paraguay. It is important to remark that it was the first application of this type of evaluation in the public sector in Paraguay. A full validation of TME requires a long-term study, including its applications in a consistent number of e-government projects. This is out of the scope, and the possibility, of the current thesis. Nevertheless, the initial results of the different case studies indicate the feasibility and simplicity of TME when it is applied in this context.

Keywords

Service-oriented computing, interoperability, developing countries, agile methods, service-oriented methods, evaluation, case studies

[1]Tape Mbo'e means: "method" in Guarani Language

Contents

List of Figures

List of Tables

Chapter 1

Introduction y Motivation

The Information and Communication Technologies (ICT) is not generally spread out across all public agencies in developing countries. This situation has entailed many pitfalls such as inconsistency and duplication of data, dishonest practices, bureaucratic procedures, and overpopulation of public employees. Thus, various governments, such as Egypt [2], Chile [3], Mozambique [4], and Paraguay [5], have implemented an interoperability framework to overcome the isolation of their agencies.

The interoperability has normally been implemented with the adoption of the service-oriented computing (SOC) technology which is a growing trend for modelling a huge class of heterogeneous applications which can belong to different agencies to provide value services to citizens. This is not always straightforward due to the constituent legacy systems have generally been developed in an independent fashion, and they could use incompatible technology. These issues can do unmanageable the development processes, unless they have a software engineering method which defines what to do, how to do, and when to do.

In these countries, the political and economical conditions impose constraints in the execution of projects. The political changes can occur suddenly retarding or cancelling projects execution, and they can also affect the staff assigned to the projects [6]. Moreover, the projects are normally financed by international donors, but a risk is their abandonment when the help finishes [7, 8]. Another constraint is the lack of either qualified or trained staff in government [9]. All these constraints affect software sustainability that is the capability of bankrolling of the software operation in long-term [8]. For these reasons, the software have to deliver fast, maintain their costs low, and consider from the beginning, aspects related to long-term sustainability. These issues seem to suggest the adoption of Agile Methods (AMs) convenient, owing they can rapidly adapt in changing environments, provide fast delivery, and focus on user satisfaction [10, 6]. However, they offer little or no documentation making software maintenance difficult, if their developers leave the organization or the development has been outsourced [11, 12]. As a result, the software become non sustainable [8]. In addition to this, they do not encompass the technical aspects of SOC which can facilitate to face the interoperability issues. For this reason, it seems convenient to take into account the Service-Oriented Methods (SOMs).

Besides, the e-government projects can involve stakeholders and staff of different agencies, and also they can be demanded by donors to fulfil quality standards. Thus, they have to deal with manage human resources, scope, costs, time, quality, etc., which are usually considered under the umbrella of project management and quality assurance. Both knowledge areas are poorly defined in AMs, being less considered in SOMs.

As a result, this thesis proposes the extension of the AM "Open Unified Process (OpenUP)[1]" to manage the development and the maintenance of service-based e-government applications. The proposal also include the project management and the quality assurance.

[1]http://epf.eclipse.org/wikis/openup/

1.1 Challenges of development methods for e-government: The case of Paraguay

We tackle the pitfall of a method to develop interoperable and sustainable e-government systems in developing countries. The interoperability is implemented through SOC, but the methods available for this technology seem to be no enough mature in the project management; they do not take into account the user satisfaction; and only a few of them has been applied in real cases [16, 17, 18, 19, 20, 21].

The major risk in these countries is that the software projects are abandoned after a first success which is a failure of sustainability [8]. The failure is caused by many factors, among them, we mention withdrawing of benefactors fund, lack of political will, scarcity of both economic resources, and qualified professionals [7, 9, 8].

We illustrate this situation using as example Paraguay where the software projects are typically supported by benefactors whose help can encompass economic resource and technicians. The legacy systems of public agencies do not share information each other, and only a few public services are available in web-sites [9]. Consequently, there are inconsistency and duplication of data that have promoted dishonest practices. Additional to this, citizens have to follow bureaucratic procedures that might involve distant agencies to access to the majority of the public services. Moreover, a survey carried out in 11 public agencies (among them are 6 ministries) revealed that the development is normally outsourced by the agencies (see Appendix B). The ICT staff of the agencies can comprehend from 5 to more than 50 people, and a software project can involve various teams located in different places. The interviewees reported that there had not defined a development method in the agencies, as a consequence, the development was managed informally, and without documentation, hindering the software sustainability.

Therefore, this situation has boosted us to find a method to develop and maintain sustainable service-based e-government applications, and our proposal is described in the next chapters.

1.2 Objectives of our research

The major goal of this thesis is the proposal of a method for the development and the maintenance of e-government applications, based on the service-oriented paradigm, in developing countries. This goal tackles various secondary goals which are the following:

- Our approach should describe the software architecture, should propose strategies to maintain the development cost low, and should be easy to apply.
- Our approach should encompass the whole life cycle of the software and should consider the quality of the development process.
- Our approach should manage development, human resources, communication, and procurement.
- Finally, it should include the service description, the service implementation, and exchange standards.

1.3 The proposal

The proposal is to extend the OpenUP to include mainly the SOC features and the operation and retirement of the software. Thus, our approach is called Tape Mbo'e (TME)[2].

TME encompasses the major characteristics of SOC, and includes the quality assurance; both are out OpenUP's scope. It has also redefined the project management discipline of OpenUP according to

[2]Tape Mbo'e means: "method" in Guarani Language

the standard of project management [22]. These changes have affected the definition of the disciplines, the roles, and the work-products included by OpenUP.

Moreover, TME have enlarged the life cycle of OpenUP, in order to overcome the gap between software delivery and its maintenance.

As a result, TME is a service-oriented method that includes the whole life cycle of software, from its conception until its retirement.

We are aware that the changes proposed by TME reduce its agility. However, they support the sustainability, which is more valuable for the type of projects that we are interested in, such as the government projects.

1.4 Structure of the Thesis

The remainder of the thesis is structured as follows:

- **Chapter 2** summarizes the **State of the Art** related to our work which are the software engineering methods. Among the methods, we are interested in Agile Methods and Service-Oriented Methods which are examined according to a set of goals. The goals involve four dimensions which are Service Oriented, Life Cycle Coverage, Project Management, and Sustainability.

- **Chapter 3** outlines the proposal of TME which is an extension of the agile method OpenUP to include the service-oriented paradigm benefits and others complementary dimensions. The description involves the life cycle, the roles, the disciplines and how the disciplines are performed during the life cycle. Moreover, the comparison between OpenUP and TME is provided.

- **Chapter 4** depicts the evaluation process that involves the phases of planning, execution, and analysis. The planning describes the evaluation goals, the metrics, and the measurement process. The execution encompasses the case studies description, their measuring, and their evaluation. Finally, in the analysis compares the outcomes of the case studies, and it discusses the main findings.

- **Chapter 5** summarizes the major contributions of this thesis to the state of the art, and proposes the conclusions. Besides, it is suggested some fields in which TME can be extended.

Chapter 2

State of the Art

As mentioned in the Introduction, service-oriented approaches have been adopted in different countries for the design and development of e-government (as well as others domain applications) with interesting results. The agile methods (AMs) also seem very convenient. In effect, they have the ability to adapt fast in changing environments, and to focus on customer satisfaction, as well as to provide value deliveries to stakeholders from early stages of development For this reason, we are interested in finding methods that include the characteristics of service-oriented computing and maintain to some extent agile approach. In addition to this, considering the constraints of developing countries, the potential approaches should also take into account aspects of project management and involve the maintenance of software. As a result, the analysis embraces four dimensions which are Service Oriented, Life Cycle Coverage, Project Management, and Sustainability. The dimensions of Service Oriented and Sustainability have determined the types of methods which will be reviewed.

Thus, we have inspected the Agile Methods and the Service-Oriented Methods, and our assessment is organized as follows. Section 2.1 depicts how the assessment is carried out. Section 2.2 presents a briefly description of some Agile Methods, and then evaluates them. Section 2.3 describes some Service-Oriented Methods which are also evaluated. Finally, conclusion is provided.

2.1 Evaluation Planning

To examine the methods we have adapted an evaluation framework which was previously used to evaluate Agent Oriented Methodologies [23]. This framework is flexible to evaluate any type of methods through the change of its goals (attributes).

Our assessment focuses on answering four research questions which guide our analysis. Each research question is associated to one dimension which is a set of characteristics related to one knowledge area. The dimensions comprehend a set of goals to be measured in order to quantify their achievement. We defined four dimensions: Service Oriented, Life Cycle Coverage, Project Management, and Sustainability. The Service Oriented and the Life Cycle Coverage dimensions are related with technical-methodological aspects. Complementary, the project management and the sustainability take into account the needs of the software development in developing countries.

2.1.1 Definition of Research Questions

We propose the following research questions:

Q.1 Does the methods include the service-oriented computing characteristics (services description, composition, identification, and service level agreement (SLA))? .

Q.2 Does the method cover the whole life cycle of service-based applications (SBAs) (requirement, analysis and design, development, and maintenance)? .

Q.3 Does the method address activities for project management?.

Q.4 Does the method support the sustainability of the software?.

2.1.2 Definition of Goals to Evaluate

The definition of goals is performed applying Goal-Question-Metric (GQM) [24]. GQM separates the goals in sub-goals hierarchically as a tree of goals where the goals-leaves measure the goals' achievement. The tree is organized in the next way: dimensions and goals, as it is shown in Table 4.1. Thus, each node of the tree is described as follows.

SO Service Oriented

This dimension focuses on the intrinsic characteristics of service-oriented computing, and it is composed of the following goals:

SO.1 *Service Description.* Finds out if the methods describe the services messages, interfaces, flows, roles, etc.

SO.2 *Service Level Agreement.* Ascertains if the methods comprehend the Service Level Agreement (SLA).

SLA depicts service, terms, guarantees, and responsibilities into contract among a service provider and their consumers [17].

SO.3 *Service Identification.* Examines if the methods propose techniques to find the functionalities to be deployed as services.

SO.4 *Service Composition.* Looks into if the methods encompass the service composition.

The service composition is to cluster various services to achieve a goal. There are four type of composition models: choreography, orchestration, coordination, and assembly [17].

LC Life Cycle Coverage

This dimension is in charge of finding out which phases of the life cycle are covered by the methods. The life cycle encompasses the following phases:

LC.1 *Requirement.* Specifies what the software has to do and how it is supposed to work [17].

LC.2 *Analysis and Design.* Examines how the architecture of the software is designed and documented. [20].

LC.3 *Construction.* Ascertains how the construction of the software is carried out [17].

LC.4 *Deployment and Provisioning.* Looks into if the methods include the activities of deployment and publication of services.

LC.5 *Operation and Management.* Ascertains how maintenance (corrective, perfective, and adaptive) and adaptation of the software is carried out.

Adaptation is the capability of changing the software to achieve new requirements or address new environment conditions after it was deploying [17].

Table 2.1: Tree of Goals

SO Service Oriented

- SO.1 Service Description
- SO.2 Service Level Agreements
- SO.3 Service Identification
- SO.4 Service Composition

LC Life cycle coverage

- LC.1 Requirement
- LC.2 Analysis and Design
- LC.3 Construction
- LC.4 Deployment and Provisioning
- LC.5 Operation and Management

PM Project Management

- PM.1 Integration management
- PM.2 Scope Management
- PM.3 Time Management
- PM.4 Cost Management
- PM.5 Quality Management
- PM.6 Human Resource Management
- PM.7 Communication Management
- PM.8 Risk Management
- PM.9 Procurement management

SU Sustainability

- SU.1 Accessibility
- SU.2 Validity
- SU.3 Vendor Independence
- SU.4 Degree of Prescription

PM Project Management

Involves the fundamental activities to manage software process. It considers the goals proposed by project management standard [22] which are the following:

PM.1 *Integration management.* Inquires about planning, developing, and controlling of the development.

PM.2 *Scope Management.* Examines if the methods define, verify, and control the development scope. In addition to this, it verifies the creation and definition of work-breakdown structure (WBS).

PM.3 *Time Management.* Ascertains if the methods estimate and control the activities duration.

PM.4 *Cost Management.* Appraises if the methods estimate costs, elaborate budget, and control the costs.

PM.5 *Quality Management.* Examines if the quality is considered and controlled. Moreover, it inquires about the verification and validation of the software.

PM.6 *Human Resource Management.* Finds out if the methods include the definition of staff' roles and their responsibilities.

PM.7 *Communication Management.* Ascertains how the methods address the communication among the stakeholders.

PM.8 *Risk Management.* Examines if the methods consider the identification of risks, and if it plans how to respond to and to recover from them.

PM.9 *Procurement Management.* Finds out if the methods consider purchases and outsourcing.

SU Sustainability

This dimension concentrates on the aspects that can affect the sustainability of the SOMs. It is composed of the following goals:

SU.1 *Accessibility.* Looks into if the methods are described with guides, examples, and case studies. Moreover, it examines if these information can be obtained without additional cost [18], [21].

The major pitfall of proprietary approaches is that their information can be unreachable.

SU.2 *Validity.* Ascertains if there have been reported some implementation of the methods in real projects or case studies. It is important for demonstrating the practical applicability of the methods.

SU.3 *Vendor Independence.* Examines if the methods are proprietary approach.

SU.4 *Degree of Prescription.* Examines if the methods provide guidelines or processes to develop software.

2.1.3 Definition of Measures

Two types of measurements are performed: direct (D) and indirect (I). The former is applied for leaves while the latter for non-leaves of the tree of goals. For each type were established the following measures: Direct and Global. They are described in Table 4.2. The Global Measure encompasses a range of values whose interpretation is the degree of satisfaction of the goals achievement. The scales of the measures are given in Table 2.3. The relations among measure, scale, and values are shown in Figure 2.1.

The value for a non-leaf is the arithmetic mean calculated from its child nodes, and it is in the Global scale. The arithmetic mean ($\bar{X}$) is obtained with Formula 4.1.

$$\bar{X} = \frac{\sum_{k=0}^{n} a_i}{n} \qquad (2.1)$$

$\bar{X}$: *Arithmetic mean*
a: *Child node*
i: *Depth of the child node in the tree*
n: *Number of child nodes*

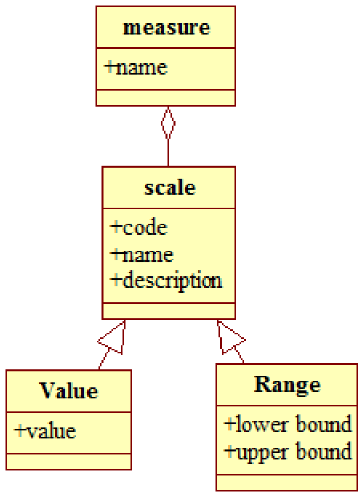


Figure 2.1: Measure Description

Table 2.2: Measures Description

Measure	**Direct**	**Global**
Code	D	G
Description	Used to measure the leaves of the tree	Applied to non-leaves and it is the arithmetic mean of the child nodes
Property Measured	Characteristic	Characteristic
Type Scale	Likert	Likert
Scale Value	Positive Numbers	Positive Numbers
Interval Value	[0,4]	[0,4]

2.1.4 Measurement of Goals

The tree is measured starting from the leaves until the root. Thus, first of all it is assigned a value in the direct Scale to the leaves (see Table 2.3). Then, the values of the non-leaves are calculated through the arithmetic mean of their child nodes. The arithmetic mean is computed with Formula 4.1, and is in the Global scale.

2.1.5 Analysis of the methods

The analysis is addressed from two perspectives: by individual goals; and by dimensions.

2.2 Agile Methods

The Agile Methods (AMs) propose a set of practices to develop software which are in accordance with the Agile Manifesto [25, 26]. The Agile Manifesto promulgates a set of practices which give value to individuals and interaction instead of processes and tools; working software instead of comprehensive documentation; customer collaboration instead of contract negotiation; and responding to changes instead of following a plan [26].

The systematic literature review carried out by Abrantes and Travassos [27], identified eighteen characteristics of AMs. These characteristics are being incremental, being cooperative, being iterative, time-boxing, leanness, people-orientation, being collaborative, transparency, self-organization, emergence, reflection and introspection, feedback incorporation, modularity, convergence, small teams,

Table 2.3: Scale Description

		Measures		
Code	**Name**	**Direct**	**Global**	**Description**
NAP	**Does Not Apply**	0	$0 \leq \bar{\mathbf{X}} < 1$	The evaluation can not be performed
NA	**Does Not Address**	1	$\bar{\mathbf{X}} = 1$	It was never achieved
VL	**Very Low**	1,62	$1 < \bar{\mathbf{X}} \leq 1{,}75$	It was achieved less or equal to 43.75% of the time
L	**Low**	2,2	$1{,}75 < \bar{\mathbf{X}} \leq 2{,}5$	It was achieved between 43.75% and 62.5% of the time
M	**Medium**	2,8	$2{,}5 < \bar{\mathbf{X}} \leq 3{,}25$	It was achieved between 62.5% and 81.25% of the time
H	**High**	3,4	$3{,}25 < \bar{\mathbf{X}} < 4$	It was achieved more than 81.25% and less of 100% of the time
C	**Complete**	4	$\bar{\mathbf{X}} = 4$	It was achieved at 100% of the time

constant testing, and local teams. Additional to this, the review determined fifteen agile practices which are coding standards, collective code ownership, continuous integration, metaphor, on-site customer, planning game, product backlog, project visibility, re-factoring, simple design, short releases, daily meetings, and sustainable pace. This review involves the general characteristics of AMs, but not describes them.

For this reason, we have examined various well known AMs which are Scrum [28, 29], Extreme Programming (XP)[30, 31], Open Unified Process (OpenUP)[1], Feature Driven Development (FDD) [32], and Dynamic Systems Development Method (DSDM) [32]. The characteristics of these AMs are summarized in Table 2.4, and they are briefly described in the next sub-sections.

Table 2.4: Summary of Agile Methods

Agile Methods	**Phases**	**Work products**	**Roles**
Scrum	• Pre-game • Development	• Product Backlog List • Sprint Backlog List • Burn Down Chart	• Core roles – Product owner – Team – Scrum master • Range of ancillary roles – Stakeholders – Managers

Next Page

[1] *http://en.wikipedia.org/wiki/OpenUP*

Table 2.4 – *Next Page*

Agile Methods	Phases	Work products	Roles
Extreme Programming	• Exploration • Planning • Iterations to Release • Productionizing • Maintenance • Death	• Story Card • Schedule • Functional Test	• Programmer • Customer • Tester • Tracker • Coach • Consultant • Big boss
Open Unified Process	• Inception • Elaboration • Construction • Transition • Operation	• Risk List • Work Items List • Iteration Plan • Project Plan • Glossary • Vision • System Wide Requirement • Use Case Model • Use Case • Architecture Notebook	• Basic Roles – Stakeholder – Project Manager – Analyst – Architect – Developer – Tester – Any • Deployment Roles – Course Developer – Deployment Engineer – Deployment Manager – Product Owner – Technical Writer – Trainer • Environment Roles – Process Engineer – Tool Specialist

Next Page

Table 2.4 – *Next Page*

Agile Methods	Phases	Work products	Roles
Feature Driven Development	• Develop of Overall Model • Built a Features List • Plan by Feature	• Features List • High-level Plan • Develop of Overall Model • Built a Features List • Plan by Feature	• Key roles – Project manager – Chief architect – Development manager – Chief programmer – Class owner – Domain experts • Supporting roles – Release manger – Language lawyer – Build engineer – Toolsmith – System administrator • Additional roles – Tester – Deployers – Technical writers
Dynamic Systems Development Method (DSDM)	• Feasibility Study • Business Study • Functional Model Iteration • Implementation	• Feasibility Report • Outline Plan • Business Area Definition (Business Object Model, ER-Diagram) • System Architecture Definition • Prototyping Plan • Configuration Management Plan • Functional Model • Prioritized Functions • Functional Prototyping Review Documents • Non-Functional Requirements List • Risk Analysis of Further Development • Training • User Manual • Project Review Report	• Executive sponsor • Visionary • Ambassador user • Advisor user • Project manager • Technical coordinator • Team leader • Solution developer • Solution tester • Scribe • Facilitator

2.2.1 Scrum

Scrum is iterative and incremental method which is composed of three phases: pre-game, development, and post-game [28, 29].

Its iteration is called **Sprint** which last as maximal one month either work finishes or not. The requirement to be developed is chosen from a priority list at the begin of the spring; and then it is developed, and validated before its delivery.

Scrum suggests the team have to arrange a meeting every day which lasts as maximal 15 minutes to update to the staff about work progress [25]. its team is composed of not more than 10 developers, and it involves two types of roles: core and ancillary roles. The core roles are product owner, team, and scrum master while the ancillary role are stakeholders and managers (see Table 2.4).

2.2.2 Extreme Programming

Extreme Programming (XP) defines a life cycle composed of the phases of Exploration, Planning, Iterations to Release, Productionizing, Maintenance, and Death [30, 31].

Over the **Exploration Phase**, the users describe in a story-card the requirement which has to be included in the first release. The sequences in which the requirements will be built are planned in the **Planning Phase**. The **Iterations to Release Phase** embraces various iterations before the first release. The Productionizing Phase verifies and validates the software before its release to users. The **Maintenance Phase** is to keep running the software while parts of it are built. The **Death phase** involves the final release of the software.

XP involves technical practices which are Incremental Planning, Small Releases, Simple Design, Test-first Development, Re-factoring, Pair Programming, Collective Ownership, Continuous Integration, Sustainable Pace, On-Site Customer, Uniform Coding Standard, and System Metaphor. They are based in the values of communication, simplicity, feedback, and courage.

XP's team involves the roles of programmer, customer, tester, tracker, coach, consultant, and big boss (see Table 2.4). The team's size must not be more than 12 people.

XP can fail in distributed teams, and when the work place is noisy [31, 33].

2.2.3 Open Unified Process

Open Unified Process (OpenUP) (version: June 1, 2012) [2] is a light version of Rational Unified Process (RUP), a well organized method that has been widely used in the industry. OpenUP encompasses various characteristics of RUP such as life cycle, iterative, and incremental process; it is use case driven, architecture centric, and risk focused. However, its iterations last generally a few weeks, and it describes only essential content either informal or formal way. Moreover, OpenUP likes RUP is organized in two dimensions: life cycle and disciplines. The life cycle is composed of four phases: Inception, Elaboration, Construction, and Transition (see Figure 2.2). The **Inception Phase** covers

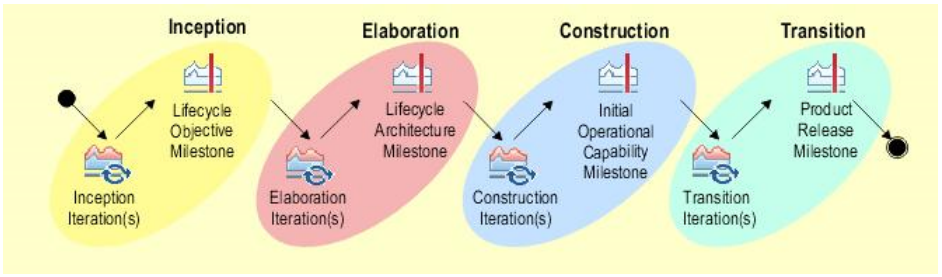

Inception
Lifecycle Objective Milestone
Inception Iteration(s)
Elaboration
Lifecycle Architecture Milestone
Elaboration Iteration(s)
Construction
Initial Operational Capability Milestone
Construction Iteration(s)
Transition
Product Release Milestone
Transition Iteration(s)

Figure 2.2: OpenUP life cycle

main functionalities identification, project plan elaboration, and requirements acquisition. The **Elaboration Phase** encompasses problem understanding, and architecture building. The **Construction Phase** embraces system constructing and testing. Finally, the **Transition Phase** involves the system delivery and deployment.

Besides, the disciplines are Requirements, Architecture, Development, Deployment, Environment and Project Management which produce work-products (see Table 2.4).

[2] *http : //en.wikipedia.org/wiki/OpenUP*

The **Requirements Discipline** focuses on understanding the problem to be solved, and the stakeholders needs. Its work-products are use cases and scenarios representing the requirements.

The **Architecture Discipline** describes the architecture through simple diagrams or list of decisions, and its work-product is the architecture notebook.

The **Development Discipline** builds the software incrementally according to the requirements, and it checks if the software accomplishes the specifications.

The **Deployment Discipline** plans the release deployment.

The **Environment Discipline** configures process and tools to be used in the development.

The **Test Discipline** verifies software in isolated and integrated manner to get early and frequent feedback in order to ensure the software achieves the requirements.

The **Project Management Discipline** focuses on addressing the project activities including management project staff, enhancing the relationships with external teams and resources, risk management, estimate scheduling and planning, managing the iteration.

OpenUP defines three type of roles which are Basic, Deployment, and Environment. The Basic roles is composed of analyst, architect, developer, any role, tester, stakeholder, and project manager. The Deployment roles includes course developer, deployment engineer, deployment manager, product owner, technical writer, and trainer. The Environment roles involves process engineer and tool specialist (see Table 2.4).

2.2.4 Feature Driven Development

Feature Driven Development (FDD) proposes five sequential processes which focus on designing and building software. The processes are Develop and Overall Model, Build a features List Plan by Feature, Design by Feature, and Build by Feature [32].

The Design by Feature and the Build by Feature are iterative processes which support the agile development.

FDD concentrates in quality aspects of the processes; and it monitors the project progress.

FDD defines three categories of roles: key roles, supporting roles, and additional roles. The key roles are: project manager, chief architect, development manager, chief programmer, class owner and domain experts. The supporting roles comprehend release manger, language lawyer, build engineer, toolsmith and system administrator. The additional roles include tester, deployers and technical writers (see Table 2.4).

2.2.5 Dynamic Systems Development Method

Dynamic Systems Development Method (DSDM) comprises five phases: Feasibility Study, Business Study, Functional Model Iteration, Design and Build Iteration, and Implementation.

The phases of Functional Model Iteration, Design and Build Iteration, and Implementation are iterative and incremental while the others are executed only once. Each phase produces work-products which are lodged in Table 2.4 [32].

DSDM is suggested for teams composed of not more than 6 developers. However, it can be used to develop large systems, with the condition of the systems have to be separated in parts that are in charge of small teams.

DSDM defines the roles of executive sponsor, visionary, ambassador user, advisor user, project manager, technical coordinator, team leader, solution developer, solution tester, scribe, and facilitator.

2.2.6 Analysis of the Agile Methods

The measurement has been performed according to the settled evaluation process described in Section 2.1.

Due to these AMs are not under the umbrella of Service-oriented computing, our evaluation does not include the service oriented dimension.

The direct and indirect measurement are presented in the next sub-sections.

Analysis of the Goals: Direct Measurement

This analysis inspects how each Agile Method (AM) fulfils the goals settled.

LC Life Cycle Coverage

This dimension is measured through the following goals:

LC.1 Requirement

The AMs normally focus on gathering requirements from users. The users choose the requirement that will be developed over short period of time. Consequently, these AMs fulfil the "Requirement goal".

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	4	4	4	4	4	Complete	Complete	Complete	Complete	Complete

LC.2 Analysis and Design

Scrum and XP do not specify any description of architecture, as a result they obtained "Does Not Address".

FDD outlines the architecture at high level then its score "Low".

OpenUP and DSDM delineate partially in narrative way the architecture; thus, they achieved "Medium".

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	1	1	2,8	2,2	2,8	Does Not Address	Does Not Address	Medium	Low	Medium

LC.3 Construction

The AMs focus on building software, and they aim at producing high quality of code. Thus, they achieve completely this sub-goal.

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	4	4	4	4	4	Complete	Complete	Complete	Complete	Complete

LC.4 Deployment and Provisioning

The responsibility of these AMs is until delivery and installation of the software.

The publication of services is out of the scope of them owing they are not under the umbrella of service-oriented computing.

Therefore, all of them have "High" as grade.

Method	SCRUM	XP	OpenUP	FDD	DSDM	SCRUM	XP	OpenUP	FDD	DSDM
Grades	3,4	3,4	3,4	3,4	3,4	High	High	High	High	High

LC.5 Operation and Management

After delivering the software, the AMs do not consider its correction, improvement, or adaptation (i.e its maintenance), as well as its abandonment. Therefore, all of them have "Does Not Address" as grade in this sub-goal.

Despite of the fact that XP involves the phases of Maintenance and Death, they do not concerned to the maintenance as it is defined above. The Maintenance Phase lasts until the software is completely finished, and the Death Phase refers at the final delivery.

Method	SCRUM	XP	OpenUP	FDD	DSDM	SCRUM	XP	OpenUP	FDD	DSDM
Grades	1	1	1	1	1	Does Not Address	Does Not Address	Does Not Address	Does Not Address	Does Not Address

PM Project Management

This dimension tackles the goals related with project management.

PM.1 Integration Management

At the beginning of the development, the AMs gather the requirement to be developed.

Scrum describes the project in high level in its product backlog including requirements and priorities. The sprint backlog plans the spring, and burn down reports about the project progress (see Table 2.4).

XP set the order in which the requirements will be built in over its Planing Phase.

OpenUP arranges the project and the iterations, as a result it elaborates the project and iteration plan.

FDD and DSDM outline the project plan. Moreover, DSDM elaborates the plans of prototyping and configuration management.

Method	SCRUM	XP	OpenUP	FDD	DSDM	SCRUM	XP	OpenUP	FDD	DSDM
Grades	2,2	1,62	2,8	1,62	3,4	Low	Very Low	Medium	Very Low	High

PM.2 Scope Management

The baseline of these methods is the requirements which can be considered as the project scope, but the requirements can change continuously, and these changes are not formally controlled. On one side, SCRUM, XP, FDD, and DSDM do not create WBS, then they achieved the grade "Medium".

On the other side, OpenUP provides a work-product called "Work Items List" whose template is published in the website of these method (see Table 2.4). The template comprises data of Name / Description, priority, size estimate, state, target iteration, assigned to effort estimate left (hours), hours worked, reference material. As a result, OpenUP have the grade "High".

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	2,8	2,8	3,4	2,8	2,8	Medium	Medium	High	Medium	Medium

PM.3 Time Management

All these methods determine the duration of tasks, and they control the accomplishment of the time settled.

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	4	4	4	4	4	Complete	Complete	Complete	Complete	Complete

PM.4 Cost Management

The estimation and control of the costs, as well as the creation of budget are not taken into account by these methods.

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	1	1	1	1	1	Does Not Address	Does Not Address	Does Not Address	Does Not Address	Does Not Address

PM.5 Quality Management

All these AMs emphasis on the quality of code as well as the verification and validation of the software. However, SCRUM, XP, FDD give scarce documentation, and they obtain the grade “Low”.

OpenUp and DSDM produce documents that not only boost the probability of accomplishing the quality norms, but also they are a memory aid to support the maintenance. Moreover, the development and test tasks of OpenUp are better defined and documented than in DSDM.

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	2,2	2,2	3,4	2,2	2,8	Low	Low	High	Low	Medium

PM.6 Human Resource Management

The roles and responsibilities are well defined in all these methods, but recruitment, task assignation, and control are informal.

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	2,2	2,2	2,2	2,2	2,2	Low	Low	Low	Low	Low

PM.7 Communication Management

The communication is one of the pillar of the AMs, but AMs do not issue documentation to support the decisions, agreements, etc.

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	3,4	3,4	3,4	3,4	3,4	High	High	High	High	High

PM.8 Risk Management

The risks are managed by SCRUM, OpenUP, FDD and DSDM, but not by XP that has the grade “Does Not Address”. OpenUP and DSDM include work-products to describe the risks, for this reason they have the grade “Complete” in this goal (see Table 2.4).

Besides SCRUM and FDD achieved the grade “High”.

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	3,4	1	4	3,4	4	High	Does Not Address	Complete	High	Complete

PM.9 Procurement Management

Neither these of AMs deal with “Procurement Management”, as a result, they have grades “Does Not Address”.

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	1	1	1	1	1	Does Not Address	Does Not Address	Does Not Address	Does Not Address	Does Not Address

SU Sustainability

This dimension encompasses the attributes related with sustainability.

SU.1 Accessibility

The AMs have gained the interest not only in the academia, but also in the industry. Thus, we can find a lot of information about them in Internet, and they also have their own website which are Scrum [3], XP[4], OpenUP[5], FDD [6], and DSDM [7]. Consequently, they all fulfil this goal.

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	4	4	4	4	4	Complete	Complete	Complete	Complete	Complete

SU.2 Validity

These methods have widely been validated not only in the academia, but also in the industry.

[3]https://www.scrum.org/
[4]http://www.extremeprogramming.org/
[5]http://epf.eclipse.org/wikis/openup/
[6]http://www.featuredrivendevelopment.com/
[7]http://www.dsdm.org/

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	4	4	4	4	4	Complete	Complete	Complete	Complete	Complete

SU.3 Vendor Independence

All these methods are open source, then they have the grade "Complete".

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	4	4	4	4	4	Complete	Complete	Complete	Complete	Complete

SU.4 Degree of Prescription

OpenUP provides more work-products than the other methods, then it has the grade "Complete" (see Table 2.4). It is followed by DSDM and FDD; DSDM has the grade "Hight" and FDD has the grade "Low".

Besides, the less number of work-products is provided by Scrum and XP, and then they have "Very Low".

Method	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Grades	1,62	1,62	4	1,62	2,8	Very Low	Very Low	Complete	Low	Hight

Analysis of the Dimensions: Indirect Measurement

The dimensions are measured indirectly through the application of the Formula 4.1. As a result, the best grade was obtained by OpenUP, and it is in the range of "Medium". However, Scrum, XP, FDD, and DSDM only obtained "Low".

These results are shown in Table 2.5 and Figure 2.3.

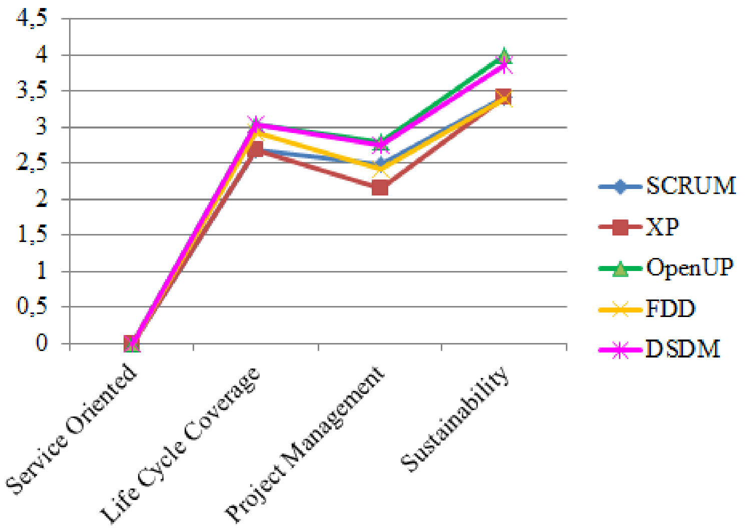


Figure 2.3: Comparison among the dimensions of the Agile Methods

The analysis of each dimension can be summarized as follows.

Table 2.5: Assessment of Agile Methods

		Grades					Interpretation				
Goals	**Evaluation Type**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**	**SCRUM**	**XP**	**OpenUP**	**FDD**	**DSDM**
Global Evaluation	**I**	**2,86**	**2,75**	**3,28**	**2,92**	**3,21**	**M**	**M**	**H**	**M**	**M**
LC. Life cycle coverage	**I**	**2,68**	**2,68**	**3,04**	**2,92**	**3,04**	**M**	**M**	**M**	**M**	**M**
LC.1 Requirement	D	4	4	4	4	4	C	C	C	C	C
LC.2 Analysis and Design	D	1	1	2,8	2,2	2,8	VL	VL	M	L	M
LC.3 Construction	D	4	4	4	4	4	C	C	C	C	C
LC.4 Development and Provisioning	D	3,4	3,4	3,4	3,4	3,4	H	H	H	H	H
LC.5 Operation and Management	D	1	1	1	1	1	NA	NA	NA	NA	NA
PM. Project Management	**G**	**2,47**	**2,14**	**2,8**	**2,41**	**2,74**	**L**	**L**	**M**	**L**	**M**
PM.1 Integration management	D	2,8	2,8	2,8	2,8	2,8	M	M	M	M	M
PM.2 Scope Management	D	2,8	2,8	2,8	2,8	2,8	M	M	M	M	M
PM.3 Time Management	D	4	4	4	4	4	C	C	C	C	C
PM.4 Cost Management	D	1	1	1	1	1	NA	NA	NA	NA	NA
PM.5 Quality Management	D	2,2	2,2	3,4	2,2	2,8	L	L	H	L	M
PM.6 Human Resource Management	D	2,2	2,2	2,2	2,2	2,2	L	L	L	L	L
PM.7 Communication Management	D	3,4	3,4	3,4	3,4	3,4	H	H	H	H	H
PM.8 Risk Management	D	3,4	1	4	3,4	4	H	NA	C	H	C
PM.9 Procurement management	D	1	1	1	1	1	NA	NA	NA	NA	NA
SU. Sustainability	**G**	**3,41**	**3,41**	**4**	**3,4**	**3,85**	**H**	**H**	**C**	**H**	**H**
SU.1 Accessibility	D	4	4	4	4	4	C	C	C	C	C
SU.2 Validity	D	4	4	4	4	4	C	C	C	C	C
SU.3 Vendor Independence	D	4	4	4	4	4	C	C	C	C	C
SU.4 Degree of Prescription	D	1,62	1,62	4	2,2	3,4	VL	VL	C	L	H

Life Cycle Coverage Dimension

The major pitfall of these methods are that they do not encompass the scarce of documentation, and they do not encompass the Operation and Management phase. Moreover, they do not embrace the service publication, and Scrum and XP do not comprise the Analysis and Design Phase.

The best grade in this dimension has been reached by OpenUP.

Project Management Dimension

The project management is generally applied in short iterations that may be documented or not in these AMs. In addition, the activities of tracking, and controlling projects can be informal [34, 35].

The goals of "Cost Management" and "Procurement Management" are not addressed by neither of these methods. Moreover, the "Time Management" is the most achieved goal for all these methods.

The most highest score in this dimension has been reached by OpenUP.

Sustainability Dimension

OpenUP has completely fulfilled the goals of this dimension. The next best grade is (i.e "High") was reached by DSDM.

The goal of "Documentation" is the weakness of the Scrum and XP, then, they has achieved "Very Low". Moreover, FDD has achieved "Low" in this goal.

2.2.7 Overview of the Agile Methods

The insufficient documentation is a problem with AMs which focus on code instead of producing documentation. Thus, they could not meet the requirements of quality standard such as ISO certificate, and Capability Maturity Model and Integration CMMI [36]. Moreover, the knowledge gets lost when either the developers leave the organization or system development has been outsourced. The large and/or complex systems maintenance, and the new members team training can be difficult or impossible without documentation [35, 11].

The analysis phase is not included by XP and SCRUM. In addition to this, neither of the AMs involves the maintenance phase.

AMs normally organizes the project activities in short iterations, but may not be documented. The project monitoring and control is not always be well-organized. Additional to this, other areas do not covered by AM are quality assurance, and configuration management [34]. The weakness in the quality control mechanisms is critical in software development where software failures can injure human or cause economic damage [35].

Besides, AMs require customers side-to-side with developers who should participate in daily meeting to be updated about the project progress. This practice requires some level of IT maturity that should be hard to find n developing countries. Moreover, it can not be applied by distributed teams, which are the common situations for e-government applications.

2.3 Service Oriented Methods

Various researches have focused on the service oriented methods (SOMs) such as Kohlborn et al [16], Andrikopoulos et al [17], Kontogogos and Avgeriou [18], Boerner and Goeken [19], Gholami et al [20], and Ramollari et al. [21].

These studies have tackled mainly the technical-methodological aspects. However, our analysis, considering the needs of developing countries, focuses also on other critical dimension which are project management and sustainability.

Table 2.6: Attributes for SOMs Analysis

Authors	**Attributes**
Kohlborn et al [16]	• SOA concept • Delivery strategy for SOA • Life cycle coverage • Degree of prescription • Accessibility and validity
Andrikopoulos et al [17]	**S-Cube Reference Life Cycle** • Analysis, Design and Implementation • Adaptation **CMMI** • Process Management • Project Management • Engineering • Support Process
Kontogogos and Avgeriou [18]	• Service description • Behaviour specification • Life cycle coverage • Detail of the approach • Adaptability • Industry
Boerner and Goeken [19]	• Basic characteristics • Business aspects • Technical aspects • Economic aspects • Components of method engineering • Principles of design science research
Gholami et al. [20]	• Development Process • Modelling Language • Service-Oriented Activities • Service-Oriented Umbrella Activities • Supportive Features
Ramollari et al. [21]	• Delivery Strategy • Life cycle Coverage • Degree of prescription • Availability • Process agility • Adoption of existing processes/techniques/notation • Industrial application • Supported role(s)

All these related works examined SOMs under perspective of a set of attributes which are shown in Table 2.6.

There are common attributes among the studies, despite they may have different denomination. The similarities are listed in Table 2.7 which shows that the attribute “Life Cycle Coverage” was the only one attribute included by all the studies.

The related works analyse various SOMs, and Table 2.8 summarizes the analysed SOMs which were presented in journals, conferences or books. Those SOMs published in vendor-website are not presented in Table 2.8, because the information released in the website might be unstable. Among these SMOs the “Service Oriented Modelling and Architecture (SOMA)” seems to be the most popular [13]; followed by Erradi et al. [37], Papazoglou and Van Den Heuvel [38] (see the last column in Table 2.8). The reasons for which these SOMs were selected for assessing were not mentioned, and neither the method used to evaluate them.

Table 2.7: Common Attributes

	Studies						
Attributes	**Kohlborn et al.**	**Andrikopoulos et al.**	**Kontogogos and Avgeriou**	**Boerner and Goeken**	**Gholami et al.**	**Ramollari et al.**	**Percentage of Occurrence**
Life Cycle Coverage/Coverage of the Generic Development Life Cycle Activities	X	X	X	X	X	X	100%
Adaptability/Adaptation	X	X	-	-	-	-	33,3%
Detail of the approach/Degree of prescription	X	-	X	-	-	X	50%
Industry/Industrial application /Validity	X	-	X	-	-	X	50%
Availability / Accessibility	-	-	X	-	-	X	33,3%
Service Level Agreements/Service Level Agreement Monitoring	-	-	-	X	X	-	33,3%

Table 2.8: References

Reference	**Source**	**Kohlborn et al.**	**Andrikopoulos et al**	**Kontogogos and Avgeriou**	**Boerner and Goeken**	**Ramollari et al.**	**Lane and Richardson**	**Year**	**Percentage of Occurrence**
Adamopoulos et al. [39]	Journal	X	-	-	-	-	X	2002	33,3%
Arsanjani et al. [13]	Journal	-	X	X	X	X	X	2008	83%
Bell [40]	Book	X	-	-	-	-	-	2008	16,7%
Böhmann and Krcmar [41]	Conference	-	-	-	X	-	-	2005	16,7%
Chang and Kim [42]	Conference	X	-	-	-	-	X	2007	33,3%
Chen et al. [43]	Conference	X	-	-	-	-	-	2005	16,7%
Emig et al. [44]	Conference	-	-	-	-	X	-	2006	16,7%
Engels et al [14],[15]	Conference	-	-	X	-	-	X	2008	33,3%
Erl [45]	Book	X	-	-	-	X	-	2005	33,3%
Erradi et al. [37], [46]	Conference	X	-	X	-	X	X	2007	66,7%
Gold-Bernstein and Ruh [47]	Book	X	-	-	-	-	-	2004	16,7%
Kaabi et al. [48]	Conference	X	-	-	-	-	-	2004	16,7%
Kim and Doh [49]	Conference	X	-	-	-	-	-	2007	16,7%
Klose et al. [50]	Conference	X	-	-	X	-	-	2007	33,3%
Kohlmann and Alt [51]	Conference	X	-	-	X	-	-	2007	33,3%
Legner and Heutschi [52]	Conference	X	-	-	-	-	-	2007	16,7%
Marks and Bell [53]	Book	X	-	-	-	-	-	2006	16,7%
Meyer and Mingins [54]	Journal	-	-	X	-	-	-	1999	16,7%
Nadhan [55]	Journal	X	-	-	-	-	-	2004	16,7%
Papazoglou and Van Den Heuvel [38]	Journal	X	-	X	-	X	X	2006	66,7%
Quartel et al [56]	Conference	X	-	-	-	-	-	2004	16,7%
Sewing et al. [57]	Conference	X	-	-	-	-	-	2006	16,7%
Sneed [58]	Conference	X	-	-	-	-	-	2006	16,7%
Stojanovic et al. [59]	Conference	X	-	-	-	-	-	2004	16,7%
Winkler et al. [60]	Journal	-	-	-	X	-	-	2007	16,7%
Zhang et al. [61]	Conference	X	-	-	-	-	-	2005	16,7%
Zimmermann et al. [62]	White Paper	X	X	-	-	X	-	2004	50%

Our analysis differs from the previous works in three main aspects. First, we selected the SOMs to be analysed through the application of a set of criteria over the upshot of a systematic literature review proposed by Lane and Richardson [63]. The systematic literature review defined research questions, a search process, and both inclusion and exclusion criteria of methods. It obtained 57 SOMs and some of them were also mentioned in the related works (see Table 2.9). A brief description of the systematic literature review is presented in the next section.

Second, our analysis was addressed according to an evaluation framework that involves a measurement process, measures, scales, and formula [23, 5]. Finally, we proposed a set of attributes, including the common attributes of the related works, not just related with the technical–methodological dimension but also with project management and sustainability. Moreover, we separated into phases the common attribute of "Life Cycle Coverage" in order to get a deeper analysis.

The first step towards the analysis is the collection and the selection of SOMs which require being performed in rigorous way. We adopted a systematic literature review (SLR). For this reason, our work is based on the SLR conducted by Lane and Richardson [63]. The protocol of this SLR is composed of the following topics:

a) *Research question.* One primary question was defined, and it is "What software processes models[8] are proposed for developing SBAs (service-based applications)?". This question aims to identify the processes (methods) for developing SBAs, in which we are also interested. Thus, it retrieved the methods published until October 2009.

b) *Data source.* The papers were got from the following electronic databases:

- IEEE Explore
- ACM Digital Library
- ISI Web of Knowledge
- SpringerLink
- ScienceDirect
- Wiley InterScience

c) *Search string.* The primary question was separated into list of terms to carry out the search in the electronic databases. The logical operator "OR" was used to include synonyms for each term, and the operator "AND" to link together each set of synonyms. The string was applied to the abstracts contained in the databases, and it is the following:

("software process" or "software engineering process" or "engineering process" or "engineering methodology" or "sose framework" or soad or soma or "architecture framework" or "architecture frameworks" or "engineering methodology" or "development technique" or "development approach" or "development methodology" or "development process" or "development processes" or "development lifecycle" or "development life cycle" or "process model" or "process framework" or "development model" or "development framework" or sdlc or cmm or cmmi or 15504 or spice or 12207 or 15288) and (sba or sose or soa or "service-based" or "service based" or "service-oriented" or "service oriented" or "service-centric" or "service centric" or "service computing" or "service engineering") [63].

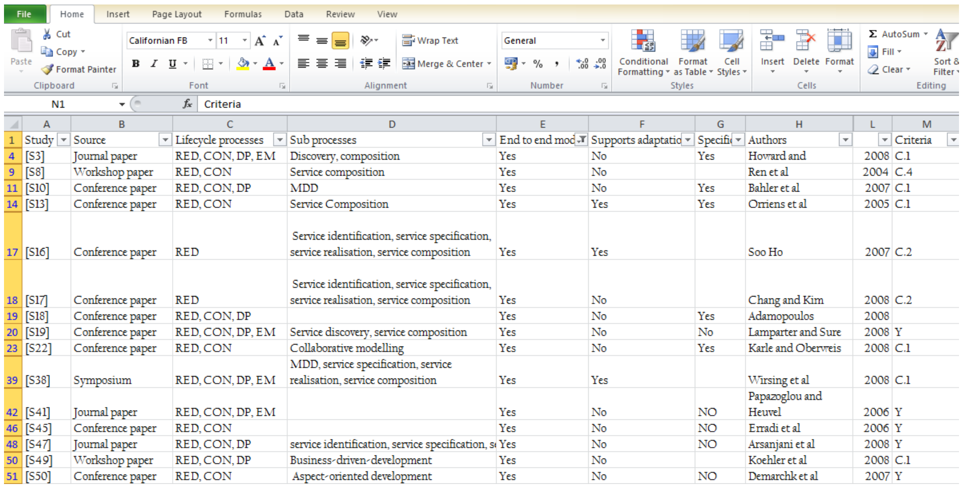
d) *Study selection.* The search retrieved 722 papers. Then the title and abstracts of the papers retrieved were read in order to filter them according to inclusion and exclusion criteria. Thus, the papers were reduced to 77 which were read completely, and then from these papers 20 were excluded again. Finally, the outcomes of SLR were 57 papers.

[8]Note that the author use "software processes models" as synonymous of methods and methodologies

e) *Data extraction.* The data extraction was supported by MS Access[9]. The papers gathered are 57, and they were published between the years 2004 and 2009. Table 2.9 shows the retrieved papers.

f) *Data synthesis.* SRL has addressed the two research questions.

We take the studies and the information given about them by the SRL to include them in Microsoft EXCEL template. Figure 2.4 shows the EXCEL template in which we can note that the first 6 columns were given in SRL's Table 5. This template was used to filter the studies again according to the following criteria:



Criteria

	A	B	C	D	E	F	G	H	L	M
1	Study	Source	Lifecycle processes	Sub processes	End to end mod	Supports adaptatio	Specifi	Authors		Criteria
4	[S3]	Journal paper	RED, CON, DP, EM	Discovery, composition	Yes	No	Yes	Howard and	2008	C.1
9	[S8]	Workshop paper	RED, CON	Service composition	Yes	No		Ren et al	2004	C.4
11	[S10]	Conference paper	RED, CON, DP	MDD	Yes	No	Yes	Bahler et al	2007	C.1
14	[S13]	Conference paper	RED, CON	Service Composition	Yes	Yes	Yes	Orriens et al	2005	C.1
17	[S16]	Conference paper	RED	Service identification, service specification, service realisation, service composition	Yes	Yes		Soo Ho	2007	C.2
18	[S17]	Conference paper	RED	Service identification, service specification, service realisation, service composition	Yes	No		Chang and Kim	2008	C.2
19	[S18]	Conference paper	RED, CON, DP		Yes	No	Yes	Adamopoulos	2008	
20	[S19]	Conference paper	RED, CON, DP, EM	Service discovery, service composition	Yes	No	No	Lamparter and Sure	2008	Y
23	[S22]	Conference paper	RED, CON	Collaborative modelling	Yes	No	Yes	Karle and Oberweis	2008	C.1
39	[S38]	Symposium	RED, CON, DP, EM	MDD, service specification, service realisation, service composition	Yes	Yes		Wirsing et al	2008	C.1
42	[S41]	Journal paper	RED, CON, DP, EM		Yes	No	NO	Papazoglou and Heuvel	2006	Y
46	[S45]	Conference paper	RED, CON		Yes	No	NO	Erradi et al	2006	Y
48	[S47]	Journal paper	RED, CON, DP	service identification, service specification, s	Yes	No	NO	Arsanjani et al	2008	Y
50	[S49]	Workshop paper	RED, CON, DP	Business-driven-development	Yes	No		Koehler et al	2008	C.1
51	[S50]	Conference paper	RED, CON	Aspect-oriented development	Yes	No	NO	Demarchk et al	2007	Y

Figure 2.4: Template used to filter

C.1 We assumed that the more relevant studies appear in journal or conference. Thus, we only included the studies from those sources. Consequently, we kept out 11 studies.

C.2 We included only those methods whose origin were the service oriented computing (SOC) or business-driven-development (BDD); being both areas closely related, and considering the indication of some studies which reported the SOC methods were bit mature [63, 21, 20]. Thus, we excluded those methods which are extensions of other paradigms such as the agent oriented development (AOD), aspect oriented (AO), and model driven development (MDD). MDD approaches comprehend collaborative modelling (CM) [91], semantic modelling (SM) [101], context aware (CA) [107], and web based (WbB) [64]. Thus, 29 approaches were left.

C.3 The methods for SOC found in the literature can focus on the construction of service, service-based applications (SBAs), or both [63]. This study encompasses those to develop SBAs which must cover at least the requirement engineering, analysis and design, and construction . Therefore, 16 studies were quit.

Table 2.9 provides a classification of all the studies in accord with these criteria.

Finally, we obtained six SOMs whose papers were read completely for analysis. These SOMs are the Service Oriented Modelling and Architecture (SOMA) [13], the Service Oriented Architecture Framework (SOAF) [37], the Service-Oriented Design and Development Methodology (SODM) [38], the Quasar Enterprise Method (QEM) [14], [15], the proposals of Demchak et al. (DEM) [78], and Lamparter et al. (LAM) [96].

These SOMs are summarized in the next subsections.

[9]https://www.zotero.org/

Table 2.9: Lane and Richardson References

Reference	Source	Origin	Coverage	Criteria	Selected
Adamopoulos et al. [39]	Conference	SOC	Service	C.5	N
Anaby-Tavor et al. [64]	Conference	WbB	Service	C.2	N
Arsanjani et al. [13]	Journal	SOC	SBA	-	Y
Arsanjani and Allam [65]	Conference	SOC	Service	C.3	N
Autili et al [66]	Conference	MDD	Service	C.2	N
Bahler et al. [67]	Conference	MDD	SBA	C.2	N
Bercovici et al. [68]	Conference	MDD	Service	C.2	N
Bicer et al. [69]	Symposium	SOC	Service	C.1	N
Bocchi et al. [70]	Workshop	SOC	Service	C.1	N
Boukadi et al. [71]	Conference	MDD	Service	C.2	N
Chang and Kim [72]	Conference	SOC	Service	C.3	N
Chang and Kim [42]	Conference	SOC	Service	C.4	N
Chengjun [73]	Conference	MDD	Service	C.2	N
Christou et al. [74]	Journal	SOC	Service	C.3	N
Colombo et al. [75]	Conference	SOC	Service	C.3	N
De Castro et al. [76]	Journal	MDD	Service	C.2	N
Delessy et al. [77]	Conference	MDD	Service	C.2	N
Demarchk et al. [78]	Conference	SOC	SBA	-	Y
Deubler et al. [79]	Conference	MDD	Service	C.2	N
Deubler et al. [80]	Conference	MDD	Service	C.2	N
Dori [81]	Workshop	MDD	Service	C.1	N
Engels et al [15], [14]	Conference	SOC	BDD	-	Y
Erradi et al. [37]	Conference	SOC	BDD	-	Y
Fernandez et al. [82]	Workshop	SOC	Service	C.1	N
Hong-Mei [83]	Journal	SM	SBA	C.2	N
Howard and Kerscgverg [84]	Journal	SM	SBA	C.2	N
Huigui et al. [85]	Conference	SOC	Service	C.3	N
Ivanyukonich et al. [86]	Journal	SOC	Service	C.5	N
Johnson et al. [87]	Conference	MDD	Service	C.2	N
Kambhampaty [88]	Conference	SOC	Service	C.3	N
Karam et al. [89]	Journal	SOC	Service	C.3	N
Karhunen et al. [90]	Conference	SOC	Service	C.3	N
Karle and Oberweis [91]	Conference	CM	SBA	C.2	N
Kenzi et al. [92]	Conference	MDD	Service	C.2	N
Koehler et al. [93]	Workshop	BDD	SBAs	C.1	N
Kruger and Mathew [94]	Conference	AO	Service	C.3	N
Kürger [95]	Workshop	MDD	Service	C.1	N
Lamparter and Sure [96]	Conference	SBA	SOC	-	Y
Lu and Chabra [49]	Workshop	AOD	Service	C.1	N
Lopez et al. [97]	Workshop	MDD	Service	C.1	N
Meisinger and Kruger [98]	Conference	SOC	Service	C.3	N
Mittal [99]	Thesis	MDD	Service	C.2	N
Nguyen et al. [42]	Journal	SOC	Service	C.2	N
Orriens et al [100]	Conference	MDD	SBA	C.2	N
Pahl [101]	Journal	SM	Service	C.2	N
Papazoglou and Heuvel [38]	Journal	SOC	SBA	-	Y
Protogeros et al. [102]	Book	SOC	Service	C.1	N
Ren et al. [103]	Workshop	SOC	SBA	C.1	N
Rychly and Weiss [104]	Conference	MDD	Service	C.2	N
Soo Ho [105]	Conference	SOC	Service	C.2	N
Strosnider et al. [106]	Journal	MDD	Service	C.2	N
Vale and Hammoudi [107]	Conference	CA	Service	C.2	N
Wada et al [108]	Conference	MDD	Service	C.2	N
Wautelet et al. [77]	Conference	AOD	Service	C.2	N
Wirsing et al. [109]	Symposium	MDD	Service	C.1	N
Wirsing et al. [110]	Conference	MDD	Service	C.2	N

2.3.1 Service Oriented Modelling and Architecture

Service Oriented Modelling and Architecture (SOMA) provides guidances, techniques, and best practices to support the development of service-based applications (SBAs) [13]. It has a fractal life cycle which is composed by seven not lineal phases: Business Modelling and transformation, Identification, Specification, Realization, Solution Management, Implementation, Deployment, Monitoring and Management.

Its major activities are Service Identification, Specification, and Realization. The **Service Identification** focuses on identifying a list of candidate services to support the development of requirements. The identification is carried out using techniques such as Goal-Service Modelling, Domain Decomposition, and Existing Asset Analysis.

The **Service Specification** selects the services to be developed from the list of candidate services, and then it describes them. The description encompasses dependencies, flows, services composition, events, rules and policies, operations, messages, non-functional requirements, and state management decisions.

The **Realization** builds prototypes and carries out feasibility exploration to verify the decisions. Moreover, it chooses pattern categories including information service patterns, enterprise service bus (ESB), and rule patterns.

SOMA is proprietary of IBM, and it has been applied in this company' projects [13].

2.3.2 Service Oriented Architecture Framework

Service Oriented Architecture Framework (SOAF) is a framework for defining, designing, and realizing Service Oriented Architecture. It provides a set of activities which are carried out during five phases. The phases are Information Elicitation, Service Identification, Service Definition, Service Realization, and Roadmap and Planning [37].

SOAF includes techniques and guidelines for identifying services, deciding service granularity, and specifying services.

Besides, SOAF embraces Service Level Agreement (SLA) and Business Level Agreement (BLA).

2.3.3 Service-Oriented Design and Development Methodology

Service-Oriented Design and Development Methodology (SODM) is an iterative and incremental methods, and it is composed of the phases of Planning, Analysis and Design, Construction, Testing, Provisioning, Deployment, Execution, and Monitoring [38].

SODM focuses on provisioning, deploying, executing, and monitoring services. It tackles various realization scenarios for business processes, and Web services.

Moreover, it depicts the details of Service Level Agreement (SLA).

2.3.4 Demchak et al.

Demchak et al. (DEM) proposes an iterative process composed of three phases: Service Elicitation, Rich Services Architecture, and System Architecture Definition [78].

The **Service Elicitation Phase** encompasses the requirements collection which are represented using use cases. From the use cases is built the domain model that includes logical entities, data structures, roles, and other knowledge representations.

The **Rich Services Architecture Phase** obtains a hierarchic set of Rich Services from the domain model.

The relationship between the Rich Services model, and its implementation is carried out in **System Architecture Definition Phase**.

2.3.5 Lamparter et al.

Lamparter et al. (LAM) encompasses the phases of Requirements Gathering, Design and Realization, and Evaluation [96].

The **Requirements Gathering** examines legacy systems and the domain of the application to be developed.

The **Design and Realization** involve the conceptualization of both the market and the ontology. The upshot of this phase are the formal model and ontology formalization.

The **Evaluation** consists on testing the application and validating against the requirements.

2.3.6 Quasar Enterprise Method

Quasar Enterprise Method (QEM) is a business driven process for designing and realizing services. It is composed of four processes which are Modelling business services, Designing domains, Designing components, Designing interfaces and operations [14, 15].

The **Modelling business services** aims at identifying top-level business services, identifying services actions, refining business services, and specifying business services.

The **Designing domains** structures IT architecture according to business services. It defines the activities of apply core business services, apply business dimensions, apply business objects, apply management and support service, and finalize.

The **Designing components** comprehends the activities of assigning services to domains, categorizing services, refining components, and verifying coverage of components.

The **Designing interfaces and operations** builds the interfaces and operation of the components. Besides, it validates the operations completeness.

QEM has been applied in industrial projects [15].

2.3.7 Analysis of the Service Oriented Methods

The measurement has been performed according to the settled evaluation process described in Section 2.1.

We have addressed the analysis from two perspectives: by individual goals; and by dimensions. Both are presented in the next sections.

Analysis of the Goals: Direct Measurement

The grades of the goals leaves are given in direct scale (see Table 2.3). This means that each grade is assigned according to the level of goal accomplishment.

The evaluation for all the goals are presented in the next sub-sections.

SO Service Oriented

This dimension focuses on the goals related with service oriented computing. The outcomes are the presented in the next subsections.

SO.1 Service Description

SOMA describes the services including flows, messages, operation, and non-functional requirements in the Service Specification activity.

SOAF depicts the services with their flows, messages, and operations.

SODM specifies services through three elements which are structure, behaviour, and policy.

DEM describes the services, hierarchic set of rich services, and relationship between service model and its implementation.

LAM specifies the services with messages, flows, and dependencies.

QEM identifies, and describes service in the step of Analysis of Business Architecture.

Methods	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**
Grades	4	4	4	4	4	4	Complete	Complete	Complete	Complete	Complete	Complete

SO.2 Service Level Agreement

SOMA, DEM, QEM, and LAM do not involve SLAs.

SODM and SOAF gather non-functional requirements, and they monitor and compel their accomplishment through SLAs [38]. Moreover, SOAF includes the Business Level Agreements (BLA).

Methods	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**
Grades	1	4	4	1	1	1	Does Not Address	Complete	Complete	Does Not Address	Does Not Address	Does Not Address

SO.3 Service Identification

SOMA proposes three techniques for identifying services: Goal-Service Modelling, Domain Decomposition, and Existing Asset Analysis.

SOAF combines top-down domain decomposition along with bottom-up application portfolio analysis to identify services.

SODM identifies and specifies service during its Analysis phase.

QEM and LAM deal with service identification as well.

DEM does not involve service discovery.

Methods	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**
Grades	4	4	4	1	4	4	Complete	Complete	Complete	Does Not Address	Complete	Complete

SO.4 Service Composition

SOMA composes service using choreography.

SOAF, SODM, and QEM describes service composition with orchestration.

LAM and DEM do not involve the service composition.

Methods	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**
Grades	4	4	4	1	1	4	Complete	Complete	Complete	Does Not Address	Does Not Address	Complete

LC. Life Cycle Coverage

We have matched the life cycle of each process with the life cycle proposed by S-Cube [17]. Thus, the results is shown in Table 2.10.

LC.1 Requirement

SOMA encompasses the requirement phase with its Identification phase that embraces activities related with Analysis Phase as well.

SOAF covers the requirement phase with Information Elicitation Phase.

Table 2.10: Relation among the phases of SOMs

Processes	Requirement	Analysis and Design	Construction	Deployment and Provisioning	Operation and Management
SOMA [13]	Identification	Business Modelling and Transformation; Identification; Specification; Realization; Solution Management	Implementation	Deployment, Monitoring, and Management	—
SOAF [37]	Information Elicitation	Service Identification; Service Definition	Service Realization	—	—
SODM [38]	Planing	Analysis and Design	Construction; Testing;	Provisioning; Deployment	Monitoring
DEM [78]	Service Elicitation	Rich Service Architecture	System Architecture Definition	—	—
LAM [96]	Requirement Gathering	Design and Realization	Design and Realization; Evaluation	—	—
QEM [15]	Modelling Business Services	Designing domains; Designing components	Designing interfaces and operations	—	—

SODM's Planning Phase carries out the activities that should performed during requirement phase while DEM performs them during the Service Elicitation Phase.

QEM covers this phase with the Modelling Business Services.

LAM includes the phase of Requirement Gathering.

Methods	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**
Grades	4	4	4	4	4	4	Complete	Complete	Complete	Complete	Complete	Complete

LC.2 Analysis and Design

SOMA covers the Analysis and Design phase with its phases of Business modelling and transformation, Identification, Specification, and Realization.

SOAF encompasses this phase with the phases of Service Identification and Service Definition while DEM do it with its phases of Rich Service Architecture and System Architecture Definition.

SODM involves this phase with its phase of Analysis and Design.

LAM includes this phase into Design and Realization, but its phase involves also the realization.

QEM embraces this phase with its phase of Analysis of Business Architecture and Definition of an Ideal Application Landscape.

Methods	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**
Grades	4	4	4	4	4	4	Complete	Complete	Complete	Complete	Complete	Complete

LC.3 Construction

SOMA includes the Implementation phase that is in charge of building, generating, and assembling services as well as components and flows.

Moreover, the unit and integration tests of services, components, and flows are performed [13].

SOAF involves the service realization phase that constructs services, adaptors, and wrappers. However, it does not consider the validation and verification activities.

SODM proposes the construction and test phases. The former involves building new web services, transforming existing application into web services, composing services, and building wrappers on top of legacy applications. The latter verifies and validates the SBA performing various types of tests such as dynamic, functional, performance, interface, and assembly testing [38].

LAM includes the Realization and Evaluation phases. The former implements the ontology while the latter is in charging of verifying and validating [96].

QEM builds and validates services [14, 15].

DME does not involve the development phase.

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	4	2,8	4	1	4	4	Complete	Medium	Complete	Does Not Address	Complete	Complete

LC.4 Deployment and Provisioning

SOMA proposes packaging, provisioning, executing user-acceptance testing, and deployment of services in the production environment during the Deployment, monitoring, and management phase [13].

SODM covers this phase with the phases of Provisioning and deployment.

This phase is not included by SOAF, DEM, QEM, and LAM.

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	4	1	4	1	1	1	Complete	Does Not Address	Complete	Does Not Address	Does Not Address	Does Not Address

LC.5 Operation and Management

Only SODM involves the maintenance during its monitoring phase, but it does not consider the service retirement.

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	1	1	3,4	1	1	1	Does Not Address	Does Not Address	High	Does Not Address	Does Not Address	Does Not Address

PM Project Management

This dimension tackles the goals related with project management.

PM.1 Integration Management

SOMA and SODM deal with planning project, but not with monitoring and controlling its execution.

The others methods do not embrace the Integration Management.

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	1,62	1	1,62	1	1	1	Very Low	Does Not Address	Very Low	Does Not Address	Does Not Address	Does Not Address

PM.2 Scope Management

Neither of the methods deal with creating work breakdown structure, monitoring and controlling scope.

SOMA, SODM, QEM, and LAM encompass the collection of requirements and definition of scope.

Moreover, SOAF and DEM comprise the requirements gathering, but not the scope.

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	2,2	1,62	2,2	1,62	1,62	2,2	Low	Very Low	Low	Very Low	Very Low	Low

PM.3 Time Management

All of methods involve a set of activities, but they do not estimate their duration, and do not control the achievement of schedule.

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	1,62	1,62	1,62	1,62	1,62	1,62	Very Low	Very Low	Very Low	Very Low	Very Low	Very Low

PM.4 Cost Management

Only SODM achieves completely this goal while SOAF and QEM do it partially.

SOMA, DEM, and LAM do not consider the cost issues.

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	1	2,2	4	1	1	2,2	Does Not Address	Low	Complete	Does Not Address	Does Not Address	Low

PM.5 Quality Management

All these methods outline a process, but neither of them encompass the quality of development process, tacking, and controlling quality.

The verification and validation are included by SOMA, SODM, LAM, and QEM, but not by SOAF and DEM.

Moreover, SOAF and SODM involve the quality of service.

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	2,2	2,2	2,8	1,62	2,2	2,2	Low	Medium	Very Low	Low	Low	Low

PM.6 Human Resource Management

These methods do not define the development roles. Thus, they comprise task assignation and controlling human resources. Consequently, they have the grade "Does Not Address".

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	1	1	1	1	1	1	Does Not Address	Does Not Address	Does Not Address	Does Not Address	Does Not Address	Does Not Address

PM.7 Communication Management

Neither of the methods manage the communication issues.

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	1	1	1	1	1	1	Does Not Address	Does Not Address	Does Not Address	Does Not Address	Does Not Address	Does Not Address

PM.8 Risk Management

SOMA is a risk-driven approach. SOAF and SODM identify risks, and SOAF also specifies a mitigation plan.

DEM, LAM, and QEM do not deal with risks.

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	4	4	1,62	1	1	1	Complete	Complete	Very Low	Does Not Address	Does Not Address	Does Not Address

PM.9 Procurement Management

Neither of the methods deal with sellers, purchasing and contracts.

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	1	1	1	1	1	1	Does Not Address	Does Not Address	Does Not Address	Does Not Address	Does Not Address	Does Not Address

SU Sustainability

This dimension encompasses the attributes related with sustainability.

SU.1 Accessibility

Various researches have focused on SOMs, we mention Kohlborn et al [16], Andrikopoulos et al [17], Kontogogos and Avgeriou [18], Boerner and Goeken [19], Gholami et al [20], and Ramollari et al. [21].

SOMA, SOAF, SODM, and QEM were mentioned in at less one of these studies, and among them, SOMA was the most popular. However, DEM and LAM were only included by the systematic literature review [63].

Thus, Figure 2.5 shows the percentages in which these methods appeared in these studies.

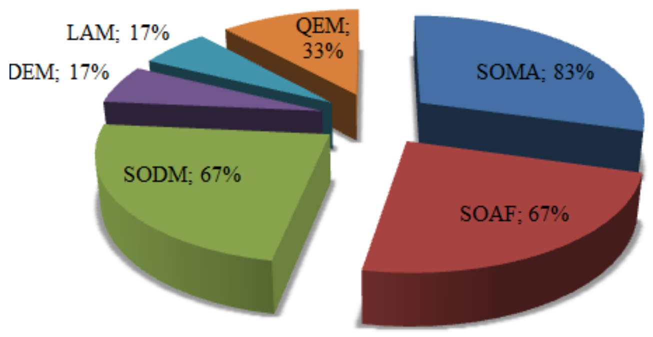


Figure 2.5: Percentage of references in related works

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	4	3,4	3,4	1,62	1,62	2,8	Complete	High	High	Very Low	Very Low	Medium

SU.2 Validity

SOMA and QEM were reported they were used in industrial environment while the others are limited to academic context [13, 14, 15].

Methods	SOMA	SOAF	SODM	DEM	LAM	QEM	SOMA	SOAF	SODM	DEM	LAM	QEM
Grades	4	1,62	1,62	1,62	1,62	4	Complete	Very Low	Very Low	Very Low	Very Low	Complete

SU.3 Vendor Independence

SOMA is proprietary of IBM while SOAF, SODM, DEM, LAM, and QEM are free to use [21], [16].

Methods	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**
Grades	1	4	4	4	4	4	Does Not Address	Complete	Complete	Complete	Complete	Complete

SU.4 Degree of Prescription

SOMA, SOAF, SODM, LAM, and QEM depicts the development process, but they do not mention the work-products.

DEM describes its process and the work-products produced by its activities.

Methods	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**
Grades	3,4	3,4	3,4	4	3,4	3,4	High	High	High	Complete	High	High

Analysis of the Dimensions: Indirect Measurement

The dimensions are measured indirectly through the application of the Formula 4.1. The best grade was obtained by SODM, and it is in the range of “Medium”. In the same range are SOMA, SOAF, and QEM with lower grades. The grades of DEM and LAM are in the range of “Low” with different values (see Table 2.11). Figure 2.6 illustrates the assessment results for all these SOMs, and it also allows us to compare them.

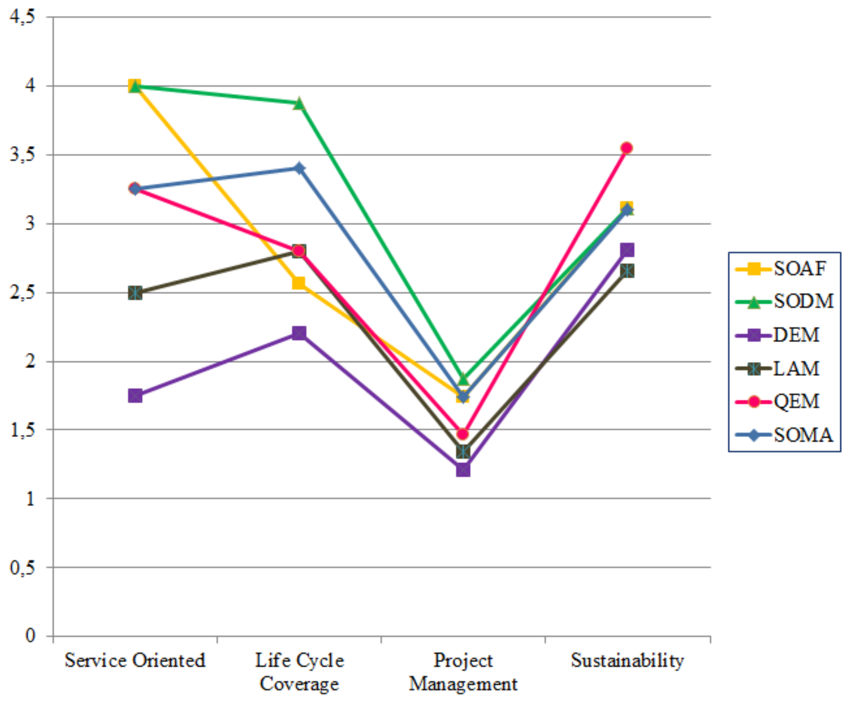


Figure 2.6: SOMs Comparison

Moreover, Table 2.11 shows the outcomes which are analysed in the next sub-sections.

Service Oriented Dimension

This dimension claims to answer the question “Q.1”, and it was fulfilled by SOAF and SODM.

Table 2.11: Assessment of Service-Oriented Methods

		Grades						Interpretation					
Goals	**Scale**	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**	**SOMA**	**SOAF**	**SODM**	**DEM**	**LAM**	**QEM**
Global Evaluation	**G**	**2,87**	**2,85**	*3,22*	**1,99**	**2,33**	**2,77**	**M**	**M**	**M**	**L**	**L**	**L**
SO. Service Oriented	**G**	**3,25**	*4*	*4*	**1,75**	**2,5**	**3,25**	**M**	**C**	**C**	**VL**	**L**	**M**
SO.1 Services Description	D	4	4	4	4	4	4	C	C	C	C	C	C
SO.2 Service Level Agreement	D	1	4	4	1	1	1	NA	C	C	NA	NA	NA
SO.3 Service Identification	D	4	4	4	1	4	4	C	C	C	NA	C	C
SO.4 Services Composition	D	4	4	4	1	1	4	C	C	C	NA	NA	C
LC. Life cycle coverage	**G**	**3,4**	**2,56**	*3,88*	**2,2**	**2,8**	**2,8**	**H**	**M**	**H**	**M**	**M**	**M**
LC.1 Requirement	D	4	4	4	4	4	4	C	C	C	C	C	C
LC.2 Analysis and Design	D	4	4	4	4	4	4	C	C	C	C	C	C
LC.3 Construction	D	4	2,8	4	1	4	4	C	M	C	NA	C	C
LC.4 Deployment and Provisioning	D	4	1	4	1	1	1	C	NA	C	NA	NA	H
LC.5 Operation and Management	D	1	1	3,4	1	1	1	NA	NA	H	NA	NA	NA
PM. Project Management	**G**	**1,74**	**1,74**	*1,87*	**1,21**	**1,34**	**1,47**	**VL**	**VL**	**L**	**VL**	**VL**	**VL**
PM.1 Integration management	D	1,62	1	1,62	1	1	1	VL	NA	VL	NA	NA	NA
PM.2 Scope Management	D	2,2	1,62	2,2	1,62	1,62	2,2	L	VL	L	VL	VL	L
PM.3 Time Management	D	1,62	1,62	1,62	1,62	1,62	1,62	VL	VL	VL	VL	VL	VL
PM.4 Cost Management	D	1	2,2	4	2,2	1	1	NA	L	C	L	NA	NA
PM.5 Quality Management	D	2,2	2,2	2,8	1,62	2,2	2,2	L	L	M	VL	L	L
PM.6 Human Resource Management	D	1	1	1	1	1	1	NA	NA	NA	NA	NA	NA
PM.7 Communication Management	D	1	1	1	1	1	1	NA	NA	NA	NA	NA	NA
PM.8 Risk Management	D	4	4	1,62	1	1	1	C	C	VL	NA	NA	NA
PM.9 Procurement management	D	1	1	1	1	1	1	NA	NA	NA	NA	NA	NA
SU. Sustainability	**G**	**3,1**	**3,11**	**3,11**	**2,81**	**2,66**	*3,55*	**M**	**M**	**M**	**M**	**M**	**H**
SU.1 Accessibility	D	4	3,4	3,4	1,62	2,2	2,8	C	H	H	VL	L	M
SU.2 Validity	D	4	1,62	1,62	1,62	1,62	4	C	VL	VL	VL	VL	C
SU.3 Vendor Independence	D	1	4	4	4	4	4	NA	C	C	C	C	C
SU.4 Degree of Prescription	D	3,4	3,4	3,4	4	3,4	3,4	H	H	H	C	H	H

All the methods included the "Service Description", and their level of achievement was "Complete".

With exception of DEM and LAM, the others encompass the "Service Composition", and the level reached was "Complete".

Apart from DEM, the others included the "Service Discovery", and their level was "Complete".

The "Service Level Agreement" was only considered by SOAF and SODM at level of "Complete".

Life Cycle Coverage Dimension

This dimension claims to answer the question "Q.2".

All the methods fulfilled the phases of Requirement and Analysis and Design phase, then they got "Complete".

The construction phase was included by SOMA, SOAF, SODM, and QEM, but not by DEM. However, SOAF did not consider the testing, and validation.

The deployment and provisioning phase was considered by SOMA and SODM, but not by SOAF, DEM, QEM, and LAM.

Only SODM embraces partially the operation and management phase, and it obtained "High".

Project Management Dimension

This dimension claims to answer the question "Q.3". It is the weakest dimension for all these methods, as we can see in Figure 2.6. Consequently, the highest level reached was "Low", and it was achieved only by SODM.

The grades of SOMA and SODM were into the range of "Very low" in the "Integration Management" goal. However, this goal was not involved in the other methods.

All the methods embraced "Scope Management" at either "Low" or "Very low" levels.

The "Cost Management" was completely fulfilled by SODM. SOAF and DEM were into the range of "Low" in this goal. However, this goal was not found in the other methods.

All of the methods slightly outline the goals of "Time Management" and "Quality Management", and their grades were in the range of "Very low".

Neither of the methods took into account the goals of "Human Resource Management", "Communication Management", and "Procurement Management". Consequently, their major weakness were the definition of roles and responsibilities as well as tracking and controlling staff, communication, contracts, and providers.

The "Risk Management" goal was completely accomplished by SOMA and SOAF whilst slightly by SODM.

Sustainability Dimension

This dimension claims to answer the question "Q.4".

The most sustainable is QEM that has obtained "High". SOAF, SODM, and SOMA are in the second place at the level of "Medium".

SOMA was used in industrial environment, and it was widely mentioned in the literature as well. Nevertheless, it is proprietary of IBM.

QEM is the only free method that reports some experience at the industry [15]. The other free distribution methods, SOAF, SODM, DEM, and LAM have been applied only in academic scenarios.

2.3.8 Overview of the Service Oriented Methods

Before giving an overview of the assessment of SOMs, it is worth noting that the validity might be constrained by the available information about the methods. Moreover, the grades of the non-leaves-goals and the dimensions depend on the leaves-goals. For this reason, the grades assigned to the leaves-goals are critical due to they determine the whole assessment, and they are constrained by the reviewer's knowledge, experience, and perception.

Despite of these considerations, we observed that neither of these methods fulfilled the settled goals.

The basic characteristics of the service-oriented computing were encompassed by all the analysed methods but not those more specific characteristics such as the service level agreement and the service composition. In addition to this, neither of these SOMs embraces the whole life cycle.

The main weakness of all these SOMs is the project management area that has to be well-defined for the development of e-government software in either developed or developing countries.

Most of these SOMs have not been implemented in real cases which is critical to validate the proposal, and ensure its practical applicability [111, 112]. Moreover, even if the methods (i.e., SOAF, LAM, DEM, and QEM) are free available, their description is normally limited to only one paper not available for public distribution. This is a remarkable difference with the methods of other more consolidated paradigms, such as object-oriented or agent-oriented which have information published even in Internet. Consequently, these aspects (i.e., validity, accessibility, and degree of prescription) reduce the sustainability of the SOMs.

Otherwise, the most interesting method seems to be SODM [38], but its major weakness is that we did not find any report of its application in real cases.

SOMA achieved the second greater assessment, and it was reported being implemented in many IBM's projects [13]. However, the information about those projects was unreachable due to SOMA is proprietary of IBM. The need of buying the SOMA's license should represent a pitfall for public institutions in developing countries.

Summarizing, despite the adoption of the service oriented computing paradigm is promising in support the interoperability, the most cited SOMs do cover adequately all the dimensions claimed as necessary for supporting the design, development, and maintenance of service-oriented e-government applications for developing countries. This results paving the way for improving the existing methods or proposing ones that can offer to modellers and practitioners the benefits of service oriented computing as well as project management aspects taking into account a sustainable perspective in the long term.

2.4 Concluding remarks on the State of the Art

We have looked for software engineering methods to develop and maintain service-based applications which can be implemented in developing countries. This context introduces additional constraints in our searching, such as the methods have to produce fast deliveries, maintain development cost low, and consider from the beginning aspects related to long term sustainability of the solutions. These issues seem to suggest the adoption of light weight methods, such as Agile Methods (AM), more convenient. However, none of the AMs examined includes the service-oriented computing characteristics (see Table 2.12). These characteristics are included by SOMs, but they are not as good as AMs in the dimensions

Table 2.12: Global Results

Methods	**Service Oriented**		**Life Cycle Coverage**		**Project Management**		**Sustainability**		**Upshot**	
Scrum	0	Does Not Apply	2,68	Medium	2,47	Low	3,41	High	2,88	Medium
XP	0	Does Not Apply	2,68	Medium	2,14	Low	3,41	High	2,75	Medium
OpenUP	0	Does Not Apply	3,04	Medium	2,8	Medium	4	Complete	3,28	High
FDD	0	Does Not Apply	2,92	Medium	2,42	Low	3,4	High	2,92	Medium
DSDM	0	Does Not Apply	3,04	Medium	2,74	Medium	3,85	High	3,21	Medium
SOMA	3,25	Medium	3,4	Medium	1,74	Very Low	3,1	Medium	2,86	Medium
SOAF	4	Medium	2,56	Medium	1,74	Very Low	3,11	Medium	2,85	Medium
SODM	4	Medium	3,88	High	1,87	Low	3,11	Medium	3,22	Medium
DEM	1,75	Very Low	2,2	Low	1,21	Very Low	2,81	Medium	1,99	Low
LAM	2,25	Low	2,8	Medium	1,34	Very Low	2,66	Medium	2,33	Low
QEM	3,25	Medium	2,8	Medium	1,47	Very Low	3,55	Medium	2,77	Medium

of project management and sustainability.

As far as project management is concerned, SOMs do not take into account the user satisfaction, and they do not define roles, these aspects are well addressed by AMs. Moreover, there is less information about SOMs than AMs, and among SOMs only Service Oriented Modelling and Architecture (SOMA) [13] and Quasar Enterprise Method (QEM) [14, 15] have been validated in real cases; these aspects are related with the sustainability.

These results seem to affirm our early hypothesis. Thus, we have chosen among AMs, the method which fits better the dimensions defined, and the method was OpenUP (version: June 1, 2012) [10]. OpenUP is the light version of the Rational Unified Process (RUP) which is a well-organized methodology and it has been widely used in the industry. It has been chosen after examining Scrum [28, 29],

[10] *http : //en.wikipedia.org/wiki/OpenUP*

Extreme Programming [30, 31], Feature Driven Development [15], Dynamic Systems Development Method [15].

Finally, the analysis of the state of the art outlines the improvements that have to be introduced in OpenUP which will be described in the next chapter.

Chapter 3

Tape Mbo'e: A Service Oriented Method

This chapter describes our proposal that is a service-oriented method called Tape Mbo'e (TME). TME was not built from scratch; it is based on Open Unified Process (OpenUP)[1], that was adapted to support the development of service-based applications (SBAs).

TME extends the life cycle and disciplines of OpenUP to cover the maintenance and retirement of applications and to assure the quality of SBAs. Besides, TME improves the project management discipline of OpenUP, and the changes have been inspired on the standard of project management [22].

The sections are organized as follows. Section 3.1 depicts the life cycle, including phases. Section 3.2 presents the kinds of roles and the roles description. Section 3.3 describes the disciplines with their work-products. Section 3.4 outlines how the disciplines are performed during the life cycle. Section 3.5 indicates the differences between TME and OpenUP. Finally, conclusion is provided.

3.1 Life Cycle

TME's life cycle involves two periods of time; the first one is the development cycle while the second one encompasses the operation of the software. This life cycle is shown in Figure 3.1.

The Development Cycle is like OpenUP's life cycle; thus, it starts with the development request and ends with the software delivery. It is iterative, incremental, and the phases are executed sequentially, as we can see in Figure 3.1. Each iteration produces an incremental value to stakeholders, results of iterations should be defined in the iteration plan. The duration of the iteration might vary depending on project characteristics, but it is suggested to last for one-month.

The increment is a unit of work that can be used to measure project progress, and to make decisions in each iteration.

To cover the gap between the software delivery and its retirement in OpenUP's life cycle, TME adds the "Operation Phase". The Operation Phase embraces from delivery until retirement of the software.

However, the development cycle has a different configuration when the development is outsourced. This configuration is represented in Figure 3.2 which shows a development cycle reduced to the phases of Inception and Transition.

The next sub-sections describe the phases whose sequences of execution are illustrated in Figure 3.1.

[1] http://epf.eclipse.org/wikis/openup/

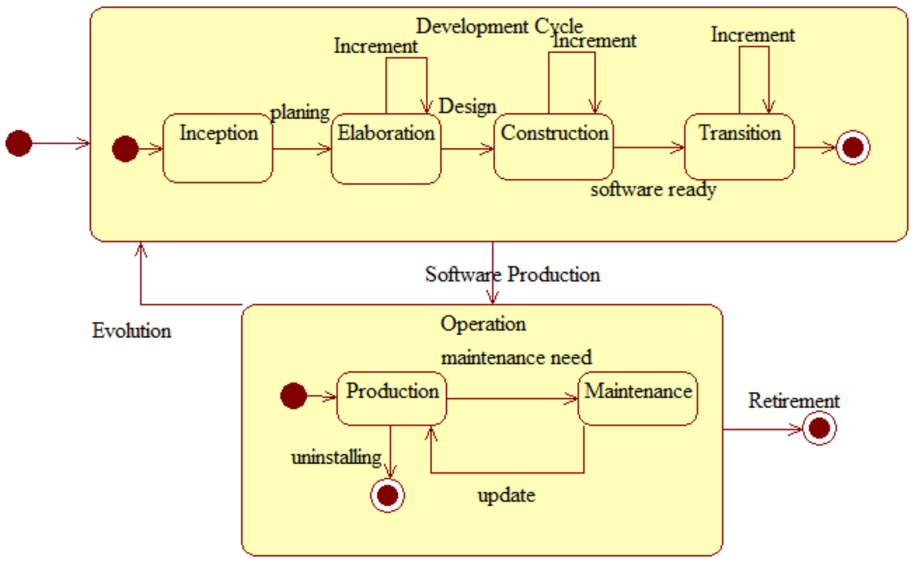


Figure 3.1: Life Cycle of in House Development

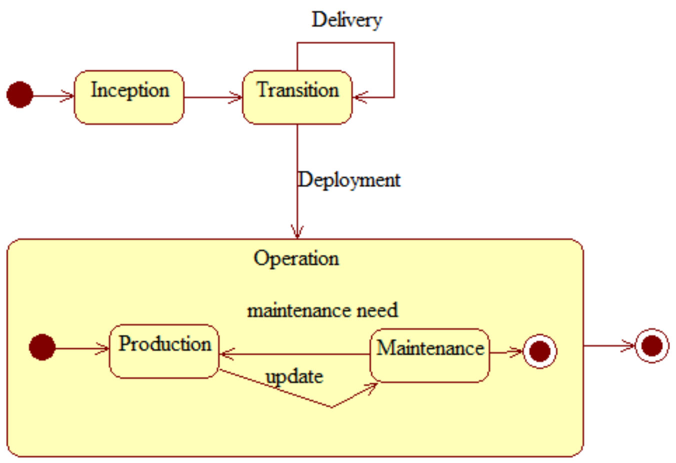


Figure 3.2: Life Cycle of Outsource Development

3.1.1 Inception Phase

The starting point of the life cycle is a development request. Thus, in this phase, it starts up a development project to attend this request.

At the first iteration, it is studied the viability of the development which determines the realization or not of it. In case the development will be carried out, the project is planned, the major requirements are collected to estimate cost, human resources, time, and to identify stakeholders. Moreover, it is taken decisions about development in-house or outsourcing, purchasing, recruitments, etc.

In the next iterations, it is selected the requirement to be developed, and the users can ask for changes in the requirements.

In this phase, it is planned and arranged the current iteration owing it is the first phase of the life cycle

In case the development is in-house, from this phase is passed to the Elaboration Phase, otherwise, the step is to the Transition Phase (see Figures 3.1 and 3.2).

3.1.2 Elaboration Phase

This phase aims at outlining the architecture of the software, described by work-products of the disciplines executed during it. It is carried out only when the development is in-house and it can involve various increments.

At the beginning, it is selected the requirement to be developed during the iteration. Secondly, the selected requirement is described considering the remark of users, and it is refined incrementally.

Moreover, staff, schedule, and risks are all managed, and the quality of the work-products are monitored and controlled.

At the end, the work-products generated are passed to the construction phase where the implementation is done.

3.1.3 Construction Phase

During this phase the software is built according to the design, and the implementation has to fulfil the pre-established quality standards.

This phase can be composed of various increments, and it is performed only when the development is in-house.

Once the software is ready, it is deployed in testing environment to be tested firstly by developer, and then by quality analyst. It is important to mention that these roles can involve various people.

At the end, the software should pass all the tests to go into with the next phase (see Figure 3.1).

3.1.4 Transition Phase

In this phase, the activities focus on testing the software with the final users, training them, and deploying the software in production environment. This phase can comprehend various increments.

At the begin, the testing timetable is defined with the final users; later the users should check the software.

In case pitfalls were detected in the software during tests or the users rejected it, the software is come back to the Construction Phase.

The users have to sign the acceptance of the software when this fulfils their requirements. Next, the software is deployed in production environment that can include the services publication in the services repository. Moreover, the users are trained on how to use the software.

At the end, the software is running in production environment.

3.1.5 Operation Phase

This phase encompasses software productive life, and during this time, the software can need to be maintained or retired from production environment (see Figure 3.1). The kinds of maintenance, which can be done, are the following:

- **Corrective** focuses on repairing faults and errors detected during the software operation
- **Perfective** aims at adding improvements in the software.
- **Adaptive** adjusts the application to support changes such as in the infrastructure layer; in the application context and location; of the user types, preferences, and constraints; in the functionalities provided by the composed services; in the way the service is being used and managed by its consumers. Furthermore, in this phase it is possible to adapt or substitute the component services to improve the performance of the software.

Once the maintenance is finished, the software goes on running.

In case the software needs to include new functionalities, it is started a new development cycle towards the software evolution.

And lastly the software can be retired from production environment, in this case the legacy systems and the services related with it have to go on working correctly.

This phase is normally the longest of the life cycle.

3.2 Roles

Tape Mbo'e (TME) defines roles, each in charge of tasks needed to meet objectives. Each role can be played by more than one person.

TME includes the three kinds of roles proposed by OpenUP[2] which are Basic, Deployment, and Environment. In addition to this, it introduces a new kind called External Roles to involve anyone that is not part of the staff.

The roles are depicted in the next sub-sections.

3.2.1 Basic Roles

This type of roles encompasses those who participate in the development and stakeholders.

TME introduces two changes in the basic roles of OpenUP, it removes the Any Role and includes Quality Analyst. The Any Role was exempted because it does not participate in a specific discipline. Moreover, the Quality Analyst was incorporated to be in charge of the Quality Assurance discipline while the user role, that is an external role, represents whom uses the software.

The dependency among the roles are shown in Figure 3.3, and they are depicted as follows.

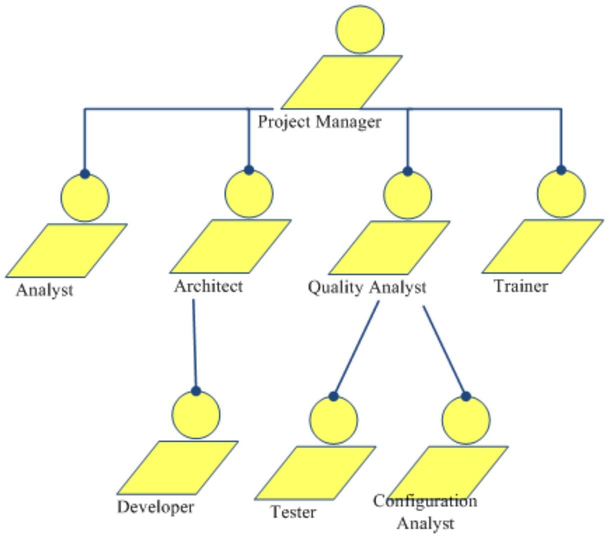


Figure 3.3: Dependency among the roles

- **Stakeholder** represents all the people interested on the execution of the project or affected by it. In this sense, all the roles can be considered as a stakeholder.
- **Project Manager** is responsible for the Project Management discipline. Its job is to direct and control development, manage human resources and relationship with providers.
- **Analyst** is responsible for collecting the requirements to elaborate use cases and mock-ups.

  This role depends on the Project Manager, and is in charge of the Requirements Discipline.

[2]http://epf.eclipse.org/wikis/openup/

- **Architect** is responsible for outlining an architecture which meets the requirements settled. It depends on the Project Manager, and is in charge of the Architecture Discipline.

  Moreover, Architect is in charge of the technical decision-making that can affect the whole development, and elaborates the work-products of the Elaboration Phase.

- **Developer** constructs the software according to the architecture and perform verification tasks once the development is done. Moreover, it carries out the maintenance of the software.

  This role depends on the architect, and is in charge of the Development Discipline.

- **Quality Analyst** is responsible for the Quality Management sub-discipline. The role is in charge of ensuring the quality of the process and deliverables.

  At the end of a phase or iteration, the Quality Analyst controls and evaluates the quality of the work products of the disciplines to suggest adjustments or improvements if necessary.

- **Tester** is responsible for verifying and validating the software and supports its revision by users.

  This role depends on the Quality Analyst, and is in charge of the Testing Sub-discipline.

- **Trainer** is responsible for training users on software usage. To accomplish this task she or he has to prepare the training materials.

  This role depends on the Project Manager.

3.2.2 Deployment Roles

These roles are in charge of the tasks that are carried out after submitting the application. TME does not include the roles of "Course Developer", "Product Owner", "Technical Writer" of OpenUP.

TME assigns the responsibilities of the "Course Developer" to the "Trainer", and it hands the responsibilities of the "Technical Writer" to the "Quality Analyst". Also, it calls "User" to the "Product Owner".

The dependency among the roles are shown in Figure 3.4, and they are depicted as follows.

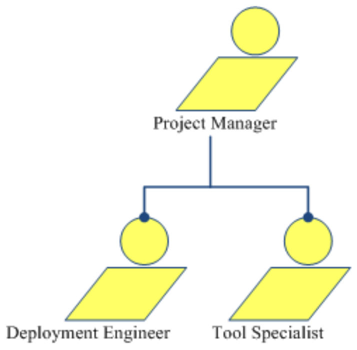


Figure 3.4: Dependency among the roles

- **Deployment Manager** is responsible for coordinating the deployment into production environment and service repository as well as monitoring and controlling the deployment engineers.

  This role is in charge of the Deployment discipline.

- **Deployment Engineer** is in charge of deploying software into production environment as well as publishing services into service repository. After deployment it verifies the success of the procedure, as well as the implementation of the roll-back plan if needed.

  This role depends on the deployment manager, and can involve various people.

3.2.3 Environment Roles

These roles include people in charge of tools, configuration, and assets. TME considers the role of "Configuration Analyst" to be in charge of the Configuration and Change Management. Moreover, it excludes the role of "Process Engineer" proposed by OpenUP because its responsibilities are carried out by the Quality Analyst.

The roles are described as follows:

- **Tool Specialist** is responsible for installing and configuring the tools to be used during the whole life cycle. It takes care of providing technical assistance about the tools used in the project including purchasing, installation, configuring, and maintenance.

  This role depends on the deployment manager, and can involve various people.

- **Configuration Analyst** is in charge of tracking and controlling the changes in the assets of the project, as well as keeping the assets.

  This role depends on the Quality Analyst, and is in charge of the Configuration and Changes Management Sub-discipline.

3.2.4 External Roles

This kind of role encompasses everybody that is not part of staff, but it is involved in the development. It is independent of anyone role and is is composed of the following roles:

- **User** is the person who asks for the development and/or will use the software.

- **Provider** is referred to people or organization which sell software.

3.3 Disciplines

A discipline is a set of tasks that have a common goal, and it aims at producing specific results, called work-products. The work-products can be documents (manual, planes, schedule, contract, etc.), models, code, templates, mock-ups, service level agreement (SLA). TME proposes templates for each work-product.

TME proposes six disciplines whose relationships are shown in Figure 3.5. The vertical disciplines (i.e., Project Management and Quality Assurance) are in charge of organizing, controlling, and suggesting tasks to the horizontal disciplines.

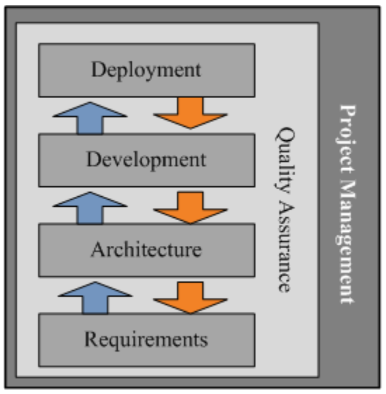


Figure 3.5: TME Disciplines

The disciplines are described in the next sub-sections.

3.3.1 Project Management

This discipline manages and controls the execution of the other disciplines. It organizes the development and handles staff, communication with stakeholders, and providers.

TME suggests using a tool for project management to support this discipline which is composed of the sub-disciplines described in the next sub-sections.

Communication Management

This sub-discipline handles the communication flow of the project, sets up the meetings frequency, and arranges the training courses for users. It makes the following work-products:

- **Meeting Minute.** Reports about subjects discussed in the meeting and the decisions taken.
- **Training Course.** Teaches to users how to use the software.

Development Management

It involves planning project activities, elaborating the project chart, estimating budget, timetable, and human resources.

Thus, it makes the following work-products:

- **Project Plan.** Depicts the project including goals, costs, timetable, milestones, human resources, and deliveries.
- **Iteration Plan.** Contains the timetable for the iteration including activities, milestones, and deliveries.
- **Interchange Agreements.** Holds the agreements and standards for data interchange among organizations or legacy systems.

Human Resources Management

It embraces the monitoring, and supporting staff to assure they accomplish their tasks.

In addition, it is in charge of training the staff to become proficient for job.

Thus, it makes the following work-products:

- **Task Assignment.** Includes the tasks assigned to the staff with their dateline.
- **Progress Report.** Describes the stage of accomplishment of the tasks, including delays.

Outsourcing Management

It negotiates with providers and it can contract them. In addition to this, it is in charge of ensuring the accomplishment of agreements, contracts and guarantees. Its work-products are depicted as follows:

- **Technical Specification.** Describes the requirements which should be present in the software to be bought.
- **Contract.** Contains the agreements between buyer and provider.
- **Supplier Tracking.** Reports about the provider's work progress.
- **Guarantee.** Contains a formal assurance about conditions will be fulfilled, especially that a product will be repaired or replaced if not of a specified quality and durability.

3.3.2 Quality Assurance

This discipline is in charge of ensuring the quality of processes and deliveries (documents, models, and software), as well as suggesting good practices. It embraces the following sub-disciplines:

Quality Management

It negotiates services quality (i.e Service Licence Agreement (SLA)). Besides, it outlines the standard processes to be followed by team, and it creates the templates for documents. Its work-products are depicted as follows:

- **Service Level Agreement.** Describes the statements about service qualities and non-functional requirements on which the service requester and the providers have reached an agreement.
- **Standards.** Comprises the definition of the templates for documents, the definition of the standards for programming and interfaces, the definition of standard processes to be followed by team.
- **User Manual.** Depicts the software and how it should be used.

Configuration and Changes Management

It focuses on tracking version over time, controlling, and managing changes in project assets. It makes the following work-products:

- **Configuration and Changes Management Plan.** Describes the process to request changes and how the changes in project assets will be tracked.
- **Changes Requests.** Contains the request of changes, who is requester, and motivation. It can be a document or being automatized by a software.
- **Version Management.** Is a software for tracking the changes in project assets.

Testing

It is in charge of verifying and validating the development. This can be carried out through different kind of tests, and it can be manual or automatized. The test cases are elaborated to be used during the verification and validation of software.

The last test is carried out with users who should accept and sign the formal acceptance of the software before its deployment.

The work-products of this sub-discipline are depicted as follows:

- **Test Case.** Describes testing scenarios for a requirement. It can be a document or being automatized by a software.
- **User Acceptance.** Is the formal acceptance of the software by user.

3.3.3 Requirements

This discipline aims at gathering the requirements of the software to be development which are outlined using use cases and mock-ups.

It arranges the requirements in a list of requirements with their priorities and development state.

Besides, it creates a glossary which is composed for those concepts whose meaning can be different through the organization in order to uniform them.

The work-products of this discipline are described as follows:

- **List of Requirements.** Is composed of all the requirements involved in the development with their priorities of development and the state of execution. It can be elaborated with any tool for office or project management.
- **Use Case.** Outlines requirements with their relations (inclusion, uses, etc.), and who carries out them. It is elaborated with the UML use case diagram.
- **Mock-ups.** Describes how the requirement will be implemented. It is made with any prototyping tool such as Pencil.
- **Glossary.** Describes the concepts used in the organization or the project.

3.3.4 Architecture

This discipline aims at outlines the architecture through the following sub-disciplines described in the next sub-sections.

Business Modelling

This sub-disciplines aims at describing processes, sub-processes, and relationships among them. It promotes the identification of services, optimization and re-utilization of processes.

Its work-product is **Business Model** which outlines a process with flow, activities, tasks, events, services, and participants. It is created with Business Process Modelling Notation (BPMN)[3] which is a standard notation.

It is elaborated one Business Model for each use case identified in the Requirements Discipline in order to outline the use case.

Integration

This sub-discipline provides a top-down description of the environment in which the application will be run including services and legacy systems. This description is contained in **Domain Model**.

It is in charge of the definition of standards to implement interoperability. Moreover, it provides the modular decomposition of the system in its **Structure Model**.

The identification of services to be re-used is another important activity. It is expected to reduce redundancy, development time, and save costs for both maintenance and operation. TME suggests the application of techniques for service identification, but does not indicate a specific one. However, some suggestions can be found in the work of Arsanjani et al [13] which are Domain decomposition, Goal-service modelling, and Existing Asset Analysis [113], [114], [115], [116]. The techniques focus on searching a set of candidate services which might achieve requirements. The search can embrace business processes, existing assets which include legacy systems, functionalities, procedures, existing services into service repository, rules and others [113].

After discovering the candidate services, they should be filtered according to pre-settled criteria such as Service Litmus Test (SLT) [13], [114]. SLT is a set of tests applied in iterative way to assess candidate services in order to select those which will be re-used.

Its work-products are described as follows:

- **Domain Model.** Outlines the services provided and consumed by legacy systems involved in the development. This is drawn with UML Component diagram according to the meta-model shown in Figure 3.6. The meta-model represents the applications as components, and the services as interfaces. Relation of dependency connects the service consumed with its component, and relation of realization links up the service provider with its component.

[3]http://www.omg.org/spec/BPMN/2.0/

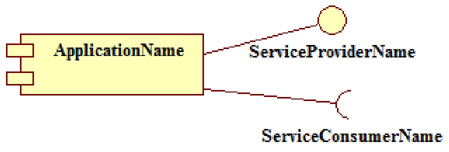


Figure 3.6: Domain Meta-model

- **Interchange Standards.** Depicts the processes to be followed to interchange data and the standard format for the data exchanged.
- **Structure Model.** Provides the modular decomposition of the system using UML component diagram.

Components Specification

This sub-discipline is in charge of providing a bottom-up depiction. Thus, it outlines each service and database in **Service Model** and **Data Model** respectively.

Its work-products are the following:

- **Data Model.** Details database structure using UML class diagram.
- **Service Model.** Depicts a service including inputs, outputs, messages, ports, and protocols according to the UML meta-model proposed by De Castro et al [1]. The meta-model is shown in Figure 3.7, and it represents the WSDL with a UML class diagram in which each part of WSDL is a class.

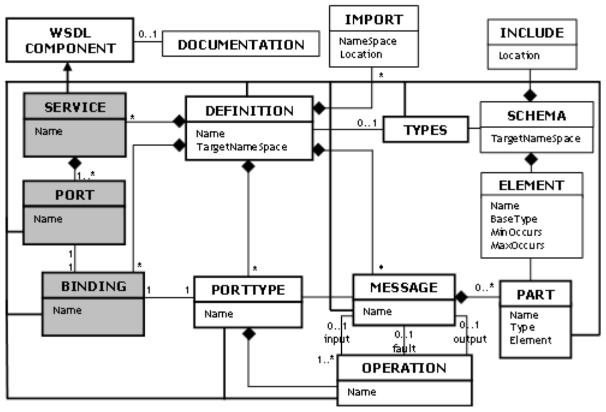


Figure 3.7: Service Model [1]

3.3.5 Development

This discipline builds the software according to the architecture specified in the models, and it implements the service composition.

Moreover, it is in charge of correcting, updating and adapting existing software as part of maintenance.

All these tasks should observe the standards defined for developing.

The work-products of this discipline are depicted as follows:

- **Software.** Is the requirements implementation.
- **Service Composition.** Carries out the orchestration of services.

3.3.6 Deployment

This discipline is in charge of install the software in either test or production environment, and it also publishes services into the service repository. The services repository is a place to keep services and the information to reach them. The industrial standards of service registry are the Universal Description, Discovery and Integration (UDDI) and Electronic Business Markup Language (ebXML) thought the most widely spread is UDDI [117].

The work-products of this discipline are described as follows:

- **Deployment Model.** Represents the run-time configuration of processing nodes, and the components executing in them with UML deployment diagrams.
- **Deployment Plan.** Reports where and when the deployment will be carried out, and who is responsible for.
- **Back-up Plan.** Depicts how the reversion of the deployment may carry out.
- **Service Repository.** Is the repository where the services are published.

3.4 Disciplines over the Life Cycle

TME is represented by analysing two dimensions: temporal evolution (life cycle) and disciplines as it is illustrated in Figure 3.8. The former shows the software life cycle, while the latter mentions the disciplines involved to produce software [5].

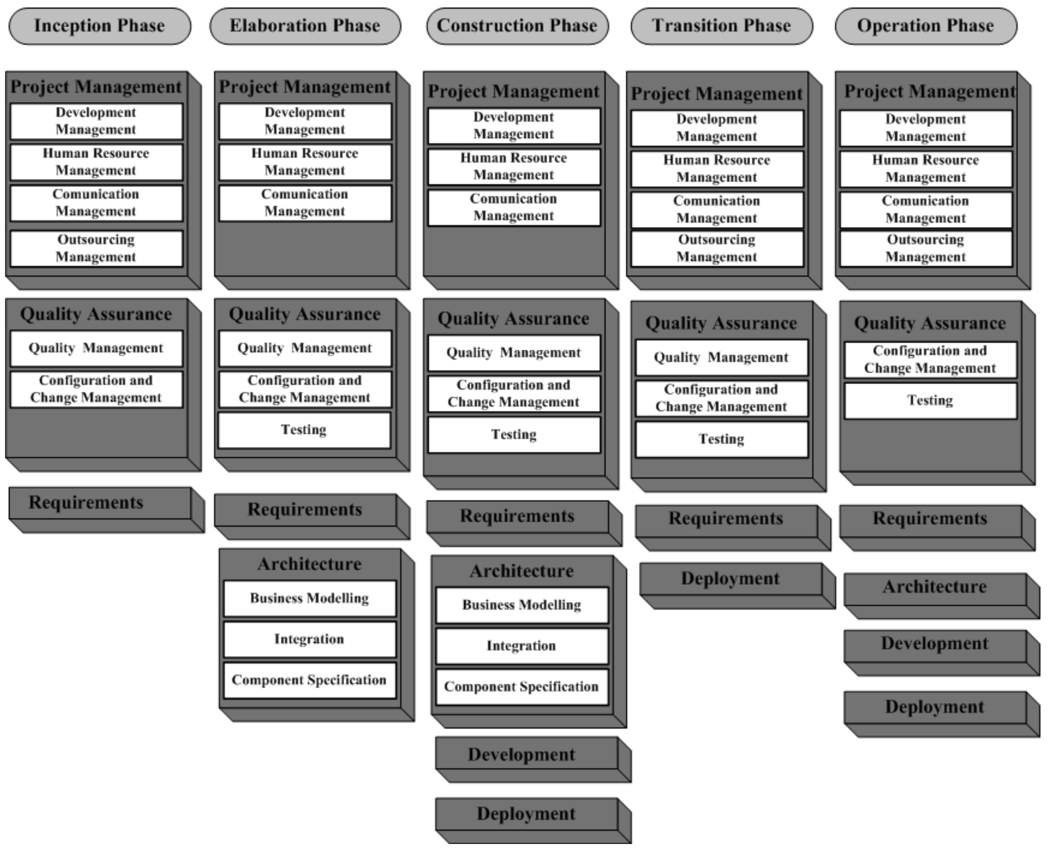


Figure 3.8: Life Cycle and Disciplines

The disciplines run through this life cycle to develop or maintain software. Table 3.1 summarizes the outcomes of the disciplines and the roles involved in them.

Table 3.1: Summary of Tape Mbo'e

Sub-Disciplines	**Work Products**	**Phases**	**Responsible Role**
Project Management			
Communication Management	• Meeting Minute • Training Course	• Inception • Elaboration • Construction • Transition • Operation	Project Manager
Development Management	• Project Plan • Iteration Plan • Interchange Agreements		
Human Resource Management	• Task Assignment • Progress Report		
Outsourcing Management	• Technical Specification • Contract • Supplier Tracking • Guarantee		
Quality Assurance			
Quality Management	• Service Level Agreement (SLA) • Standards • User Manual	• Inception • Elaboration • Construction • Transition • Operation	Quality Analyst
Configuration and Changes Management	• Configuration and Changes Plan • Changes Management • Version Management		
Testing	• Test Cases • User Acceptance		
Requirements			
—	• List of Requirements • Use Case • Mock-ups • Glossary	• Inception • Elaboration • Construction • Transition • Operation	Analyst
Architecture			
Business Modelling	• Business Model	• Inception	Architect
Integration	• Domain Model • Interchange Standards • Structure Model	• Elaboration • Construction	

Next Page

Table 3.1 – *Next Page*

Sub-Disciplines	**Work Products**	**Phases**	**Responsible Role**
Components Specification	• Data Model • Service Model	• Operation	
Development			
—	• Software • Service Composition	• Construction • Transition • Operation	Developer
Deployment			
—	• Deployment Model • Deployment Plan • Back-up Plan • Service Repository	• Transition • Operation	Deployment Manager

3.4.1 Inception

The major goals are the definition and delimitation of the development project. These are accomplished through the following disciplines:

Project Management

The project is established applying the next sub-disciplines:

Development Management. In the first iteration, a feasibility study is carried out to determine the project's viability. The project scope is delimited, and first estimations of cost, time, infrastructure, and human resources needed are done. Moreover, the risks are identified, and suitable responses to them are planned; the work-breakdown structure (WBS), schedule, and budget are elaborated as part of the project plan.

The iteration is planned independently of its number of time, it has been executed.

Human Resources Management. Recruitment can be needed in which case the candidates are interviewed, and hired thereafter.

The team is organized, and its members are notified about their responsibilities and functions.

In addition, training courses for the staff of the project are planned.

Communication Management. The stakeholders are identified and the information flow among them is prepared/organized.

Furthermore, the meeting frequency is established. Meetings with users are arranged and performed, and during those meetings, the major decisions of the project are taken.

Meeting minutes are elaborated to report on the topics discussed in these meetings

Outsourcing Management. The decision about outsourcing development is made, and as a results technical specification of purchase or engagement is created.

The suppliers are called, and their proposals are assessed in order to carry out the selection process. Finally, negotiating, awarding, and the signing of the contract with the selected supplier is done.

Quality Assurance

It aims at planning how the quality during the whole project will be managed. The following sub-disciplines are considered.

Quality Management. The definition and standardization of process and templates for documents and work-products are set. Also, good practices are suggested to be employed during the whole life cycle.

Configuration and Changes Management. It is created the Configuration and Changes Management Plan, and it is established how the change request will be addressed.

Moreover, a tool for version control is installed, if necessary. Then, it is created a directory for the project in the tool's space.

Requirements

The major project requirements are collected from stakeholders to elaborate the List of Requirements. Moreover, it is set up the priority of requirements.

Architecture

Its Integration Sub-discipline elaborates the standards to be used for interoperability. An example of a standard data format is presented in Figure 3.9 which is in XML Schema Definition (XSD). It represents the format in which the service providers present the information needed to consume their services.

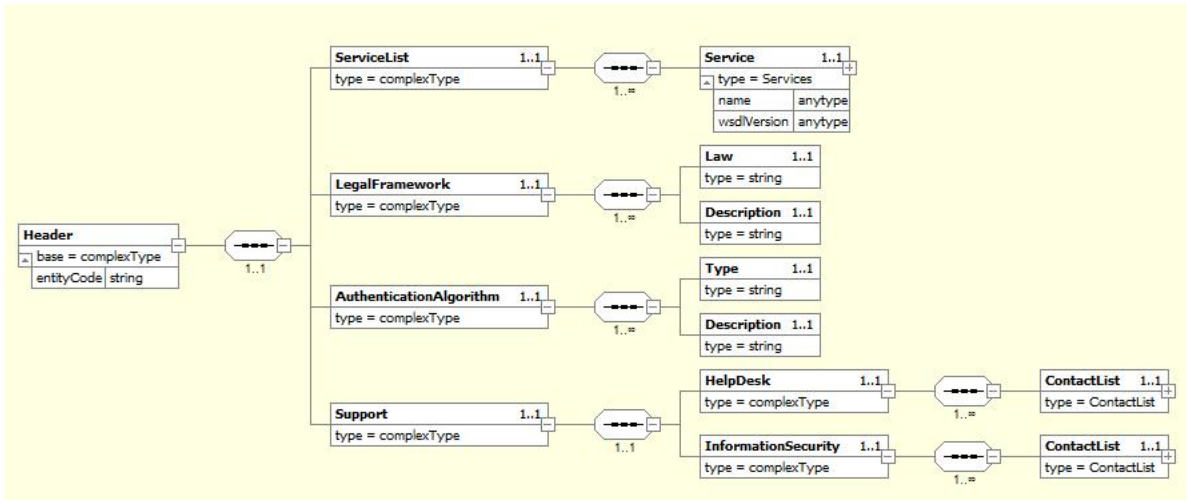


Figure 3.9: Example of Standard Data Format

3.4.2 Elaboration

This phase is only executed when the development is in-house, in which case it can involve various increments.

The disciplines executed during the phase are described in the next sub-sections.

Project Management

It is performed through the next sub-disciplines.

Development Management. At the beginning of the phase, the iteration is planned and arranged. Also the requirements to be developed during the iteration are negotiated with users.

The tasks of monitoring, controlling, and updating scope, risks, schedule, and cost are carried out.

Human Resources Management. The tasks are assigned to human resources who are monitored to ensure the tasks completion.

Communication Management. At the beginning of the phase, the analysts meet with users to determine the requirement to develop in the iteration and receive requests for changes.

Moreover, team meetings are held according to the settled frequency to update about progress, pitfalls, and support decision-making.

Quality Assurance

It focuses on the quality of design and keeping assets version. These following sub-disciplines are involved.

Quality Management. At the beginning of the iteration, the Service License Agreement (SLA) is negotiated.

The quality of the design is monitored and corrective actions can be suggested.

Configuration and Changes Management. At the first iteration, the tools to be used for designing should be installed and customized.

All the work-products created or updated have to be included in the tool used for version control.

Requirements

At the beginning of the iteration, the users should choose the requirement to be developed during it. Then, this requirement is described through use cases and mock-ups.

In case there are new terms, they are included in the Glossary.

Through the increments, these work-products can be refined and updated.

Finally, the work-products are delivered to architects.

Architecture

It is performed through the following sub-disciplines.

Business Modelling The requirement to be developed over the iteration is depicted through the Business Model. As an example the Business Model is shown in Figure 3.10. This process involves three systems: SPN, IIS and SRC. IIS gets citizens' data from SPN and SRC which then are put together.

Integration At the first iteration, the high level vision is provided in the Domain Model and Structure Model. They can be refined and updated in later increments or iterations.

As an example, the Domain Model of Information and Interchange System (IIS) shown in Figure 3.11 is proposed. The model includes the systems of the following agencies: Civil Registration Office (CROS), Ministry of Education (MES), Ministry of Heath (MHS), Secretary of Information and Communication (IIS), National Police (NPS), and Ministry of Treasury (MTS). CROS provides the *birthService* and *deathService* while NPS supplies *citizenPoliceRecordService*. IIS consumes these services, and composes them into the *citizenDataService* consumed in turn by MES, MHS, and MTS.

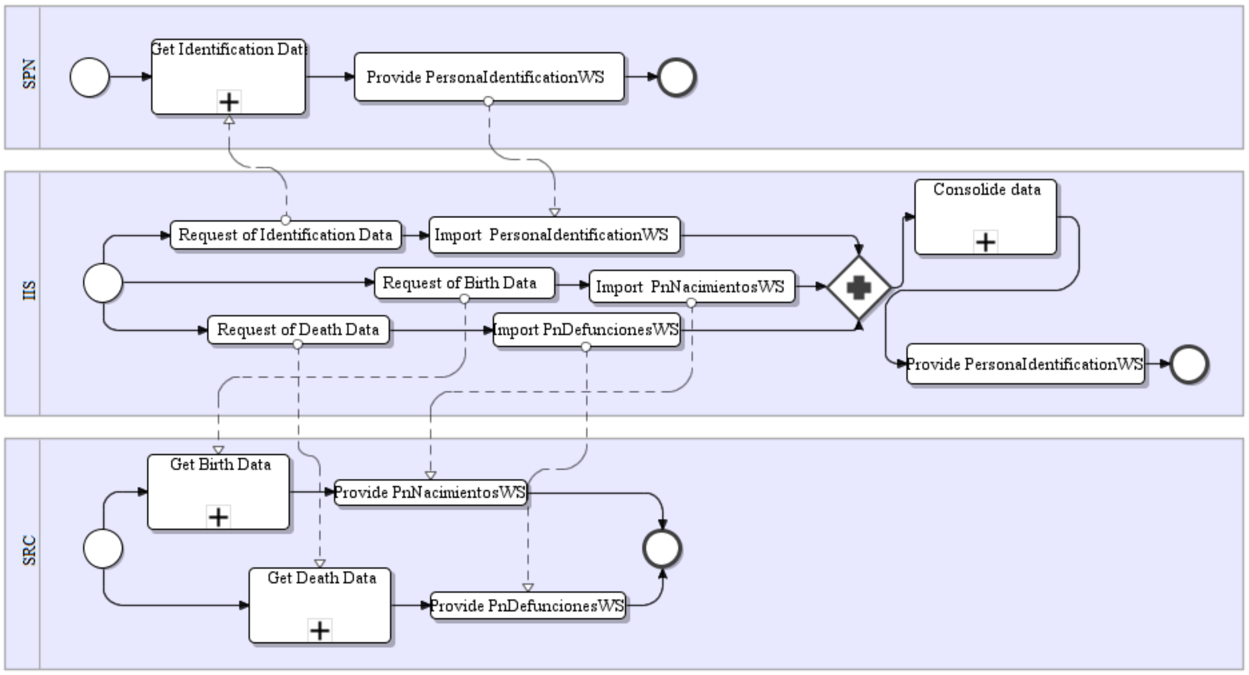


Figure 3.10: Example of Business Model

We can observe how the domain encompasses legacy systems and also produced and consumed services for them. This supports the re-usability of functionalities and services.

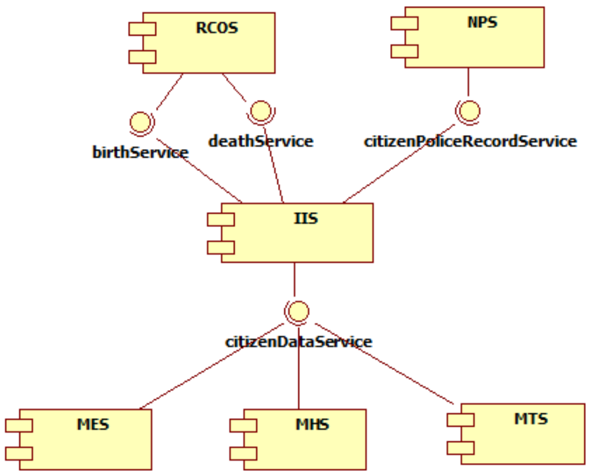


Figure 3.11: Example of Domain Model

Moreover, it is applied the techniques for candidate-services identification. Then, the candidates are filtered according to selection criteria.

Components Specification. It is elaborated the Service Models for all the services which will be developed in the iteration. The Data Model can be created or extended.

Additionally, these work-products can be refined and updated.

3.4.3 Construction

This phase is only executed when the development is in-house, in which case it can involve various increments.

During the phase, the effort is focused on building a software that fulfils the requirement. Thus, it is executed the disciplines described in the next sub-sections.

Project Management

It is performed through the following sub-disciplines.

Development Management. The tasks of monitoring, controlling, and updating scope, risks, schedule, and cost are carried out.

Human Resources Management. The human resources are monitored and controlled.

Communication Management. Team meetings are held according to settled frequency, as to update about progress, pitfalls, and support decision-making .

Quality Assurance.

It is accomplished through the following sub-disciplines:

Quality Management. The code quality is monitored, and corrective actions can be suggested.

Configuration and Changes Management. The identification, collection, classification, labelling, and maintenance of project assets version are done.

The changes of project assets are monitored, and the version is updated if needed.

Furthermore, the tools to be used for programming are installed and customized.

Testing. The types of tests to be carried out are chosen, and the test cases are elaborated. Then, the verification of functionality, usability, supportability, reliability, and performance of software is carried out by developers and testers.

Requirements

The list of requirements , use cases, mock-ups, and glossary can be updated.

Architecture

It is performed through the following sub-disciplines:

Business Modelling. The business model can be updated and refined.

Integration. Its work-products can be refined and updated.

Components Specification. The service model and the data model can be updated.

Development

It is carried out according to the design and the quality standards. The service composition is performed using an orchestration language.

Deployment

It is specified the Deployment Model, and it is elaborated the plans of both Deployment and Back-up.

Once the application is finished, it is deployed in test environment.

3.4.4 Transition

Once the software is ready to be delivered, its verification, validation, and acceptance are carried out by users. Finally, it is deployed and its productive life starts.

In this phase the following disciplines are involved:

Project Management

It is performed through the following sub-disciplines.

Development Management. The project plan and chart can be updated. Additionally, the process to authorize the application deployment is followed.

Human Resources Management. The human resources are monitored and controlled.

Communication Management. The team meeting is held to discuss about the iteration issues and lessons learned.

The analysts meet with users to verify and validate the application. Furthermore, users are trained to use the software.

Outsourcing Management. The provider delivers the application that has to be verified and validated before accepting.

Quality Assurance

It is performed through the following sub-disciplines.

Quality Management. The quality of deliverables is monitored and controlled, and improvements can be suggested.

Furthermore, the deliverable documentation and manual are elaborated to be given to final users.

Configuration and Changes Management. The deliverable version is maintained and tracked.

The change request can be received and analysed. Finally, decisions are made on whether to apply the changes or not.

Testing. The verification and validation of the software is performed by users who have to accept it before deployment.

Requirements

The list is requirements is updated.

Deployment

The infrastructure where the application will be installed is verified to ensure its adequacy, and corrective actions can be suggested. Then, the software is installed in production environment, and the services are published in the service repository.

In addition, the deployment can be reversed if necessary.

3.4.5 Operation

The activities focus on ensuring the effective and efficient performance of the software. Moreover, the software or part of it can be retired of the production environment; in this case, the operation of related applications has to be assured.

These activities are achieved with the following disciplines:

Project Management

It is performed through the following sub-disciplines.

Development Management. It deals with warranties management as well as notification of software retirement. However, a decision to start the new development iteration can be taken to evolve the software.

Human Resources Management. The staff in charge of maintenance is monitored to ensure the tasks completion.

Communication Management. The team meetings can be held.

Outsourcing Management. The warranty fulfilment can be requested to the provider.

Quality Assurance

It is performed through the following sub-disciplines.

Configuration and Changes Management. Changes on project assets are monitored, and the version is updated if necessary.

Testing. The verification and validation of the software, that have been maintained, are carried out by developers and users. Users have to accept the software before deployment.

Architecture

The business model, the domain model, the interchange standards, the structure model, and the deployment model can be updated.

Development

The corrective, adaptive, and perfective maintenance are performed.

In case, the software will be taken away from production environment, all the related legacy systems have to be examined and adjusted if necessary. Thereafter, the retirement takes place and the environment is tested again.

Deployment

Updates are deployed into production environment and / or into the service repository. The software can be returned to a previous state if necessary.

3.5 TME and OpenUP: extensions and differences

TME has introduced various changes into OpenUP, mainly, in the life cycle, roles, and disciplines. The goal of these changes has been to provide to OpenUP with the characteristics required to support the development of SBAs. Besides, they intend to overcome some weaknesses of OpenUP such as the lack of quality assurance; a life cycle disregards maintenance and retirement; project management that does not consider outsourcing, human resource, and so on.

Considering this, TME has incorporated a new phase into its life cycle "Operation Phase", in order to cover the gap between deployment and retirement of the application in OpenUP's life cycle.

On one hand, TME has added the new discipline "Quality Assurance", to assure the quality of the process and work products; and has included the Test Discipline of OpenUP into this new discipline, as a sub-discipline. On the other hand, it has given up the Environment Discipline of OpenUP, because its functions are covered by the Configuration and Change Management sub-discipline.

In addition, TME has modified various disciplines of OpenUP; the major changes have been into the Architecture and Project Management disciplines. They have been separated into sub-disciplines to specialize their management, and their work products have been incremented as well (see Table 3.3).

TME has introduced the new work product "mock-up" to the Requirements discipline. Since the service oriented computing involves a new programming paradigm, the Development discipline has been adjusted, including the service composition and the Deployment discipline to consider service publication into the service repository.

All these changes have affected the definition of roles, consequently. Four roles: Provider, User, Quality Analyst and Configuration Analyst have been added. Moreover, TME has excluded roles whose functions are covered by others, namely the: "Any", "Course Developer", "Product Owner", "Technical Writer", and "Process Engineer" roles.

The major changes incorporated in TME are shown in bold face in Tables 3.2 and 3.3.

To summarize, we highlight the improvements that TME has proposed, as follows:

- TME manages the characteristics of service-oriented computing while OpenUP does not consider these aspects. This is noted on, OpenUP's Deployment discipline missing the services deployment and publication.

- From the software engineering perspective, TME covers the whole life cycle. During its elaboration phase various work products are created to describe the application's architecture in more detail than the Architecture Notebook. The Architecture Notebook is the only work product of OpenUP that depicts the architecture (see Table 3.3).

  Moreover, TME covers a larger domain by including operations outside the scope of OpenUP

- From a quality assurance viewpoint, TME manages quality through standards, definition of process for managing configuration, controlling changes, etc. However, OpenUP does not consider these aspects.

  OpenUP involves the test practices, but does not envisage the user acceptance.

- In the project management dimension, TME proposes to manage human resource, communication, outsourcing and development as part of the Project Management discipline. By being loyal to agile philosophy, OpenUP manages the project informally, and as a result being loyal with

Table 3.2: Differences between TME and OpenUP

Item	OpenUP	TME
Life cycle	• Inception • Elaboration • Construction • Transition	• Inception • Elaboration • Construction • Transition • **Operation**
Disciplines	Project Management	**Project Management** • **Development Management** • **Human Resources Management** • **Communication Management** • **Outsourcing Management**
	Requirements Architecture	Requirements **Architecture** • **Business Modelling** • **Integration** • **Components Specification**
	Development Deployment	Development Deployment **Quality Assurance** • **Quality Management** • **Configuration and Change Management** • **Testing**
Roles	• Basic Roles – Stakeholder – Project Manager – Analyst – Architect – Developer – Tester – Any • Deployment Roles – Course Developer – Deployment Engineer – Deployment Manager – Product Owner – Technical Writer – Trainer • Environment Roles – Process Engineer – Tool Specialist	• Basic Roles – Stakeholder – Project Manager – Analyst – Architect – Developer – Tester – **Quality Analyst** – Trainer • Deployment Roles – Deployment Engineer – Deployment Manager • Environment Roles – Tool Specialist – **Configuration Analyst** • **External Roles** – **Provider** – **User**

agile philosophy. and as a result, the activities of controlling schedule compliance, cost, scope, accomplished tasks, human resources, outsourcing, and quality are not considered.

For all of these reasons, TME seems to be an interesting process in order to produce and maintain high quality service-based applications whose life cycle is supported by a well-defined project management.

3.6 Chapter Summary

Among the various Agile Methods, OpenUP has been chosen to be the baseline of Tape Mbo'e (TME). Furthermore, TME is a service oriented method that modified the life cycle and the disciplines of OpenUP. Thus, its life cycle covers the maintenance and retirement of software while the disciplines include the service-oriented computing characteristics, the quality assurance, and the improvements to the project management. These changes have affected the definition of roles and work-products as well. Consequently, TME introduces new roles, gives up some existent roles, and includes new work products.

On the one hand, TME aims at proposing a general process. For this reason, the processes for addressing more specific aspects such as the publication of services on the service repositories, techniques for service identification, criteria for reusing functionalities, metrics to measure the complexity of the development, etc. have not been included in this version of TME. However, it has been suggested the adoption of techniques proposed for another method for the service identification.

Table 3.3: Work Products of TME and OpenUP

OpenUP	**TME**
Project Management • Risk List • Work Items List • Iteration Plan • Project Plan	**Project Management** • Project Plan • Iteration Plan • **Interchange Agreements** • **Task Assignment** • **Progress Report** • **Meeting Minute** • **Training Course** • **Technical Specification** • **Contract** • **Provider Tracking** • **Guarantee**
Requirement • Glossary • Use Case Model • Use Case • System Wide Requirement	Requirements • List of Requirements • Use Case Diagram • **Mock-ups** • Glossary
Architecture • Architecture Notebook Environment • Project Defined Process • Tools	**Architecture** • **Business Model** • **Domain Model** • **Interchange Standard** • **Structure Model** • **Data Model** • **Service Model**
Development • Implementation • Build • Developer Test • Design	Development • Source Code • **Service Composition**
Deployment • Product Documentation • Support Documentation • User Documentation • Training Materials • Back out Plan • Deployment Plan • Release Communication • Release Control	Deployment • **Service Repository** • **Deployment Model** • Deployment Plan • Back up Plan
Test • Test Case • Test Script • Test Log	**Quality Assurance** • **Service Level Agreement (SLA)** • **Standards** • **User Manual** • **Configuration and Changes Management Plan** • **Change Management** • **Version Management** • **Test Cases** • **User Acceptance**

Chapter 4

Experimental Evaluation

Our goal is to assess the software development process performed according to Tape Mbo'e (TME), but not the software generated by this process. Thus, we have applied rigorous experimental evaluation to carry out this goal.

First of all, the assessment determined the goals to be measured. Then, it examined how they had been managed by the development process.

In addition to TME, we have assessed the previous organization process in order to have a reference point to compare TME.

The assessment has involved five case studies which embrace both public and private organizations in Paraguay. The breadth of the assessment is not trivial, and the assessment methodology has to manage both simple and complex situations. Among its benefits we can mention that it helps us to understand how a phenomenon is influenced by the context in which it is implemented; and the use of multiple case studies supports findings credibility [59, 42].

A full validation requires to assess not just the process but also the products. Consequently, it is a long-term study that is out of the scope of this thesis. However, early observations from this first validation process, indicated the feasibility and applicability of TME when used by practitioners.

Therefore, method used for assessment and the results of these case studies are going to be described in this chapter. It is worth noting, that this experience represented the first application of this type of evaluation in the public sector in Paraguay.

4.1 Outline of the Evaluation Process

The assessment is carried out according to an evaluation framework [5] which comprehends the phases of Planning, Execution, and Analysis. Figure 4.1 shows how the phases are organized.

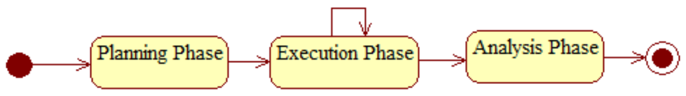


Figure 4.1: Evaluation Phases

We included into the Analysis Phase a meta-analysis method in order to compare the outcomes of various cases studies.

The phases are described in the next sections.

4.1.1 Planning Phase

This phase outlines the assessment (research questions, goals, measures, and formulas) and how the measurement will be carried out. These components are described the following.

Definition of Research Questions

The main goal is to assess the software development process performed according to TME from viewpoint of four dimensions: Service Oriented, Software Engineering, Project Management, and Sustainability. We separate this goal into secondary goals, one for each dimension. Per each secondary goal, we have introduced one research question according to Goal-Question-Metric GQM method [24]. This, let us to determine exactly what has to be measured, and the criteria to take into account to reach the goal. Thus, we have defined the following secondary goals and research questions.

- TME should include the service description, the service implementation, and exchange standards. It is called Service Oriented Dimension and its research question is:

  Q.1 Is TME able to manage the service description, the service implementation, and exchange standards, better than the previous organization process had done?

- TME should encompass the whole life cycle of the software and should consider the quality of the development process. It is called Software Engineering Dimension and its research question is:

  Q.2 Is TME able to manage the software life cycle and the quality of the development process, better than the previous organization process had done?

- TME should manage development, human resources, communication, and procurement. It is called Project Management Dimension and its research question is:

  Q.3 Is TME able to manage the development, the human resources, the communication, and the procurement better than the previous organization process had done?

- TME should describe the software architecture, should propose strategies to maintain the development cost low, and should be easy to apply. It is called Sustainability Dimension and its research question is:

  Q.4 Is TME more sustainable, (i.s it describes the architecture, provides strategies to maintain the implementation cost low, and is easy to apply), than the previous organization process?

Definition of Goals of Assessment

The secondary goals are decomposed into sub-goals hierarchically as a tree of goals (TOG) using GQM [24]. This can be repeated n-times until to reach the leaves which are the metric-associated questions used to measure the goals' achievement. Thus, in this structure, the main goal is the root of the TOG, and its value is the result of the global evaluation.

The secondary goals (dimensions) with their sub-goals are described bellow and summarized in Table 4.1. Moreover, the questions associated to each goal-leaf are shown in the Appendix A.

SO Service Oriented. Is related to the intrinsic characteristics of service computing and data interchange issues. Its major sub-goals are the following:

> **SO.1 Service Description.** Examines how well services, relationships among services, service roles, parameters, messages, service level agreements (SLA), and service composition are described.

Table 4.1: Tree of Goals

SO **Service Oriented**

- SO.1 Service Description
- SO.2 Service Implementation
- SO.3 Exchange Standards

SE **Software Engineering**

- SE.1 Life Cycle
  - SE.1.1 Requirement
  - SE.1.2 Analysis and Design
  - SE.1.3 Development and Deployment
- SE.2 Quality Assurance
  - SE.2.1 Quality Management
  - SE.2.2 Configuration and Change Management
  - SE.2.3 Test

PM **Project Management**

- PM.1 Development Management
- PM.2 Communication Management
- PM.3 Human Resource Management
  - PM.3.1 Recruitment of Human Resources
  - PM.3.2 Role Assignment
  - PM.3.3 Monitoring and Controlling Human Resource
- PM.4 Procurement management

SU **Sustainability**

- SU.1 Documentation
- SU.2 Implementation Cost
- SU.3 Easiness

SO.2 Service Implementation. Finds out how service deployment, service publication, and enterprise service bus (ESB) are carried out.

SO.3 Exchange Standards. Inquires about the processes and data format defined to support the interoperability among legacy systems.

SE Software Engineering. Involves the qualities that are required to be present in any software engineering process. Its major sub-goals are the following:

SE.1 Life Cycle. Covers the software life cycle from the conception of its original ideas until the software is abandoned. The life cycle encompasses the following sub-goals:

SE.1.1 Requirement. Inquires on how well the requirements are managed by the method according to the practitioners.

SE.1.2 Analysis and Design. Examines how the architecture of the software is designed and documented.

SE.1.3 Development and Deployment. Ascertains how the construction and deployment of the software are done.

SE.2 Quality Assurance. Focuses on the quality of development process and work products, and it is composed of the following sub-goals:

SE.2.1 Quality Management. Examines the quality of the development process; if the work-products are based on standardized templates; if good practices of development are implemented.

SE.2.2 Configuration and Change Management. Inquires how the changes are managed; if the versions of the project assets are kept up, and they are updated when necessary.

SE.2.3 Test. Inquires about the verification and validation of the software.

PM Project Management. Involves the fundamental activities necessary to manage any type of project according to project management standard [22]. Its major sub-goals are the following:

PM.1 Development Management. Encompasses the Integration Management, Scope Management, Time Management, and Cost Management of the standard of project management [22]. Consequently, it verifies how these areas of processes are managed.

PM.2 Communication Management. Ascertains how the communication is carried out among the stakeholders, and how the information is distributed.

PM.3 Human Resource Management. Verifies how staff is managed, and it includes the following sub-goals.

PM.3.1 Recruitment of Human Resources. Verifies how the processes of recruitment, selection, and hiring of human resources are done.

PM.3.2 Role Assignment. Ascertains whether the human resources know their responsibilities, and the tasks that they have to carry out.

PM.3.3 Monitoring and Controlling Human Resource. Verifies if human resources are monitored, and evaluated.

PM.4 Procurement Management. Encompasses identifying, and selecting sellers as well as awarding contracts. It also manages the relationship with sellers, while monitoring contract execution. It is referred not only to infrastructure purchases, but also to outsourcing.

SU Sustainability is the capability of organizations to support long-term software maintenance [7]. Its major sub-goals are the following:

SU.1 Documentation. Examines if the documents proposed by the method cover the needs of the organization.

SU.2 Implementation Cost. Inquires about the strategies implemented to save development costs.

SU.3 Easiness. Finds out how easy was for participants elaborate the models and document suggested by the process examined.

Definition of Measures

Two types of measurement are done: direct and indirect. The former is used for leaves while the latter for non-leaves of the tree of goals. For measures the following metrics have been established: Frequency, Aggregation of Frequency, and Aggregation of Sample. These measures are described in Table 4.2 while their scales in Tables 4.3, 4.4 and 4.5.

Table 4.2: Description of the Metrics

	Metrics		
Attribute	**Frequency**	**Aggregation of Frequency**	**Aggregation of Sample**
Code	FR	AF	AS
Description	Used to assign a value that represents how frequent a phenomenon is observed	Used to transform the frequency scale to a range of values that represent the frequency in which a goal or a dimension is achieved or not	Used to match an interval of frequencies to an interval of satisfaction
Element Measured	Questions	Goals and dimensions	Groups of samples (questionnaires)
Type of Measurement	Direct	Indirect	Indirect
Formula	No	Arithmetic mean (Formula 4.1)	Geometric mean (Formula 4.2)
Property Measured	Occurrence	Occurrence	Satisfaction
Type of Scale	Likert	Likert	Likert
Scale Value	Numeric	Numeric	Numeric
Scale	Table 4.3	Table 4.4	Table 4.5
Interval Value	[0,4]	[0,4]	[0,4]

Table 4.3: The Scales of the Frequency Metric

Meaning	**Value**	**Interpretation**
Does not Apply	0	The assessment cannot be carried out
Never	1	The measure answer has been false in all the cases
Rarely	2	The measure answer has been true less than 50% or equal to 50% of the time
Normally	3	The measure answer has been true more than 50% of the time
Always	4	The measure answer has been true in all the cases

Table 4.4: The Scales of the Aggregation of Frequency Metric

Meaning	**Range**	**Interpretation**
Does not Apply	$0 \leq \bar{X} < 1$	The goal/dimension cannot be evaluated
Never	$\bar{X} = 1$	The goal/dimension has never been achieved
Rarely	$1 < \bar{X} \leq 2{,}5$	The goal/dimension has been achieved less than 50% or equal to 50% of the time
Normally	$2{,}5 < \bar{X} < 4$	The goal/dimension has been achieved more than 50% of the time
Always	$\bar{X} = 4$	The goal/dimension has always been achieved

Measurement of Tree

The tree is measured starting from the leaves up to the root. Thus, first of all we assign a value in the Frequency Scale to the leaves (see Table 4.3), and then the value of the non-leaves is calculated.

The value of one non-leaf is the arithmetic mean calculated from its child nodes, and it is in the

Table 4.5: The Scales of the Aggregation of Sample Metric

Meaning	**Range**	**Interpretation**
Unacceptable	$\bar{G} \leq 40$	If the goals were met less than 40 % of the time, then the evaluation is unacceptable
Acceptable	$40 \leq \bar{G} < 60$	The percentage in which the goals were achieved is not too bad, then the evaluation is acceptable
Good	$60 \leq \bar{G} < 80$	The percentage in which the goals were achieved is medium, then the evaluation is good
Very Good	$80 \leq \bar{G} < 100$	The percentage in which the goals were achieved is high, then the evaluation is very good
Excellent	$\bar{G} = 100$	The evaluation fulfilled the goals

Aggregation of Frequency scale (see Table 4.4). The arithmetic mean is computed with Formula 4.1. In this way, it is measured the whole tree and it is not assigned additional weight to the goals.

However, this evaluation fashion allows to add weights to the nodes to represent priorities or importances.

$$\bar{X} = \frac{\sum_{k=0}^{n} a_i}{n} \tag{4.1}$$

a: *Child goal*
i: *Level in the tree in which the child goal is located*
n: *Number of child goals whose value in Aggregation of Frequency scale is different to zero*

X̄: *Arithmetic mean of goals measured*

Measurement of Case Study

We can take various samples for each case study, each one of them corresponding to one tree. To obtain the outcomes of the case study, we calculate the geometric mean (see Formula 4.2) among the same dimensions of the different trees. The same procedure is carried out for the global evaluation (roots).

$$\bar{G} = \sqrt[M]{R_1 * R_2 * ... * R_M} \tag{4.2}$$

Ḡ: *Geometric mean*
R: *Dimension or global evaluation*
M: *Number of trees*

Then, we translate the value of dimensions and global evaluation into the Aggregation of Sample Scale with Formula 4.3. The interpretation of the results is given in Table 4.5.

$$\bar{X} = \frac{r * 100}{max} \tag{4.3}$$

r: *Grade of Dimension or global evaluation in the Aggregation of Frequency Scale*
max: *Upper limit in the Aggregation of Frequency Scale*

S: *Value in the Aggregation of Sample Scale*

4.1.2 Execution Phase

In this phase, first we gather the samples for each of case study. Next we measure the trees and then their outcomes are used to calculate the case study results. Finally, we examine the case study results from the viewpoint of their dimensions. These steps are run for each case study and are described in the next sections.

Definition of Scope

The limits of the case study assessment are defined by both life cycle phases involved and the TOG's breadth (i.e dimensions and goals included).

Data Collection

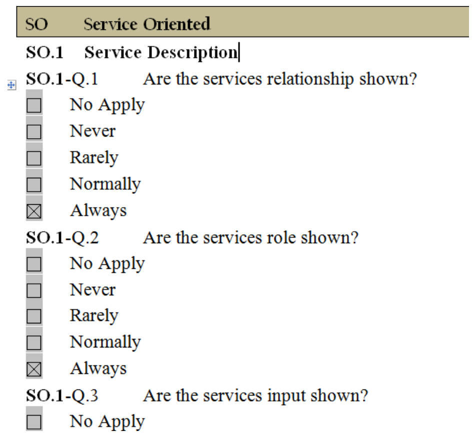
A tool was used to get the samples is a questionnaire whose questions are the leaves of the tree. The questionnaire is composed of a set of questions which correspond to the leaves of the tree of goals.

The possible responses are shown in Table 4.6. Their values are fixed for all case studies and are in the Frequency Scale (see Table 4.3). The questionnaires should be written in local language to

Table 4.6: Possible responses to one question

Responses	Value in Frequency Scale
Does not Apply	0
Never	1
No	1
Rarely	2
Normally	3
Always	4
Yes	4

simplify their interpretation. The tree of goals can be different for case studies but not for the same, for this reason, we define the scope of assessment. Figure 4.2 shows an example of questionnaire.



SO Service Oriented

SO.1 Service Description

SO.1-Q.1 Are the services relationship shown?

- ☐ No Apply
- ☐ Never
- ☐ Rarely
- ☐ Normally
- ☒ Always

SO.1-Q.2 Are the services role shown?

- ☐ No Apply
- ☐ Never
- ☐ Rarely
- ☐ Normally
- ☒ Always

SO.1-Q.3 Are the services input shown?

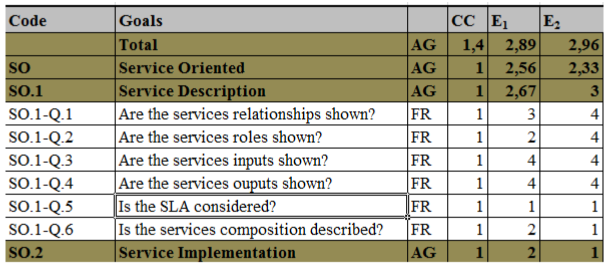
- ☐ No Apply

Figure 4.2: Questionnaire Example

Moreover, the questionnaires are classified in Control and Experimental categories, according to the process that they report on. The control questionnaires are answered by the control group, and inform about the development process before applying TME. The experimental questionnaires are answered by the experimental group and they measure TME's performance.

Measuring Process

For each questionnaire filled in, its selected questions are matched to their values in the Frequency Scale. Since the questions are the tree's leaves, from them the values are generated for the whole tree using an Excel template. This process is described in Section 4.1.1. Figure 4.3 shows the template's sheet for the example of Figure 4.2.



Code	Goals		CC	E_1	E_2
	Total	**AG**	**1,4**	**2,89**	**2,96**
SO	**Service Oriented**	**AG**	**1**	**2,56**	**2,33**
SO.1	**Service Description**	**AG**	**1**	**2,67**	**3**
SO.1-Q.1	Are the services relationships shown?	FR	1	3	4
SO.1-Q.2	Are the services roles shown?	FR	1	2	4
SO.1-Q.3	Are the services inputs shown?	FR	1	4	4
SO.1-Q.4	Are the services ouputs shown?	FR	1	4	4
SO.1-Q.5	Is the SLA considered?	FR	1	1	1
SO.1-Q.6	Is the services composition described?	FR	1	2	1
SO.2	**Service Implementation**	**AG**	**1**	**2**	**1**

Figure 4.3: Template of Measurement Example

Having generated all the trees for one case study, they are grouped according to the type of group which they refer to (i.e control or experimental).

We compute the geometric mean among the equivalent dimensions and global evaluation of different trees for each type of questionnaire. Thus, we have for each case study, the outcomes for both control and experimental groups.

Analysing the Results

The analysis is organized according the dimensions embraced by the case study, and it obtains an conclusion for each of them.

4.1.3 Analysis Phase

It combines the results of all the case studies in order to obtain a general evaluation of TME, and involves the activities described in the next sections.

Description of Meta-analysis

Meta-analysis is the statistical analysis of a large collection of outcomes from individual studies to integrate them. Among the meta-analysis method, the Vote-Counting (VCM) is applied in this study to combine the upshot of case studies [118], [119].

One advantage of VCM is that it can be applied even if the number of sample is small. VCM estimates an effect-size (∂) in order to compare the improvement magnitude of one experiment over another [120]. The effect-size can be obtained with a Weighted Mean Difference technique (WMD) like that, in Formula 4.4 [121].

$$\partial = (1 - \frac{3}{4(n^E + n^C) - 1}) * (\frac{\bar{x}^E - \bar{x}^C}{S_p}) \qquad (4.4)$$

$\boldsymbol{n^E}$***:*** *Number of experimental groups*
$\boldsymbol{n^C}$***:*** *Number of control groups*
$\boldsymbol{\bar{x}}\ ^E$***:*** *Experimental group mean*
$\boldsymbol{\bar{x}}\ ^C$***:*** *Control group mean*
S_p***:*** *Standard deviation of both experimental and control groups (see Formula 4.5)*

Formula 4.4 it is requires to compute the standard deviation (S_p), that is obtained with Formula 4.5.

$$S_p = \sqrt{\frac{(n^E - 1)(s^E) - (n^C - 1)(s^C)}{n^E + n^C - 2}} \qquad (4.5)$$

S_p: Standard deviation of both experimental and control groups
s^E: Standard deviation of experimental group
s^C: Standard deviation of control group

VCM classifies results of studies in positive, negative or non-significant. Thus, each study has a vote assigned according to the value of its effect-size which can be placed as follows:

- If ∂ is negative, the vote is -1. Then, the control group is greater than the experimental group.
- If ∂ is 0, the vote is 0. Then, both groups are equal.
- If ∂ is positive, the vote is 1. Then, the experimental group is better than the control group.

Thus, the vote's sign allows to know whether one study is better or not than another study, and the vote "0" means that none of them is better.

The effect with the highest probability of occurrence (PES) is obtained with Formula 4.6.

$$PES = \sum_{i=0}^{n} (V_i ln(1 - ((\frac{1}{S_p\sqrt{2\Pi}} e^{\frac{-1}{2}(\frac{\bar{x}^E - \bar{x}^C}{S_p})})(-\sqrt{\eta\partial}))) + (1 - V_i) ln((\frac{1}{S_p\sqrt{2\Pi}} e^{\frac{-1}{2}(\frac{\bar{x}^E - \bar{x}^C}{S_p})})(-\sqrt{\eta\partial}))) \tag{4.6}$$

***PES:** Effect size probability*
V_i: Value of case study vote
η: $(n^E + n^C)/(n^E * n^C)$
∂: Effect size to be tested
Π: Constant. Its value is 3,14159..
***e:** Constant. Its value is 2,71828..*

4.2 Execution Phase

The goal of this phase is the execution of the evaluation plan by performing the survey. TME was implemented for first time in two case studies of one university by students in order to refine the proposal and infer its simplicity owing it can be used by novice practitioners.

Next, TME was applied in three case studies which correspond to the public sector of Paraguayan government. The case studies are described in the next sub-sections.

4.2.1 Case Study 1: Academic Request System

Organization

The Catholic University (UC)[1] was set up in the fifties, and is the oldest private university of Paraguay. Its headquarters is situated in Asunción city and it has nine branches inside the country.

It was the first university in the country that included the Software Engineering career in its curriculum. The career comprehends ten semesters, and one year of thesis. The career includes in its eighth semester the course of Software Engineering which comprehends the software engineering theory. As practical part of the course, the students have to design and develop a software where they apply the practices learned during the classes.

The results of the design and development were supervised and evaluated by a professor and two assistants.

[1]http://www.uca.edu.py/

Participants

The case study was carried out by three groups composed by three students each one. This was the first experience of the students in the software design.

As far as the profile of the professor and the assistants is concerned, the professor is Ph.D. while the assistants are software engineers.

Academic Request System (ARS)

ARS aims at automating students formalities (document request, enrolments to courses or exams e.g). It has to obtain the students data from Academic System of the university through web-services. The Academic System (AS) runs in a server located in the headquarters.

ARS was installed in the server of the branch situated in Asunción city. At the beginning, ARS is going to be available for approximately 1900 students distributed across of this branch, and next for all the students of the university.

Definition of the Assessment Scope

The assessment encompasses the phases of Inception and Elaboration. Moreover, the tree of goals involved in our analysis is shown in Table 4.7. This tree excludes “Quality Assurance” and “Human Resources Management”. As a result, the assessment included 4 dimensions and 10 goals which involved 41 questions.

Table 4.7: Tree of Goals for the university' case studies

SO	**Service Oriented**
SO.1	Service Description
SE	**Software Engineering**
SE.1	Life Cycle
SE.1.1	Requirement
SE.1.2	Analysis and Design
SE.1.3	Development and Deployment
PM	**Project Management**
PM.1	Development Management
PM.2	Communication Management
SU	**Sustainability**
SU.1	Documentation
SU.2	Implementation Cost
SU.3	Easiness

Data Collection

There were two control groups, C_1 and C_2, and 3 experimental groups E_1, E_2, and E_3. The experimental groups were composed of the students while the control groups were integrated by the professor and one of the two assistants.

The control questionnaires found out about a process used by the students of another course in the previous year. The process was not service-oriented, and is different to TME. The professor and the one of the assistants completed one control questionnaire each one.

Each experimental group was composed of three students, and one of the student is the team leader. The students of each group discussed about their experience using TME in order to fill in one experimental questionnaire.

The questionnaires were sent by mail to the professor, the assistant, and the leaders of the experimental groups. After the questionnaires were completed, they were returned to the sender.

Measuring Process

First of all, the responses of all the questionnaires were included into the template which generated the results for the goal of the trees. The templates was configured according to the process defined in Section 4.1.2, and it also calculated the general results of the control groups C_{UC}, and the experimental groups E_{ARS}.

Thus, Table 4.8 presents the results for C_{UC} and Table 4.9 for E_{ARS}. These outcomes are in the Aggregation of Frequency Scale. The global results for both groups (i.e C_{UC} and E_{ARS}) are

Table 4.8: Tree of goals for the Control Groups of UC

Goals	C_1	C_2	C_1	C_2
Global Evaluation	**2,36**	**2,35**	**Rarely**	**Rarely**
SO Service Oriented	**2**	**0**	**Rarely**	**Does not Apply**
SO.1 Service Description	2	0	Rarely	Does not Apply
SE Software Engineering	**2,63**	**3,38**	**Normally**	**Normally**
SE.1 Life Cycle	*2,63*	*3,38*	*Normally*	*Normally*
SE.1.1 Requirement	2,75	3,25	Normally	Normally
SE.1.2 Analysis and Design	2,5	3,5	Normally	Normally
PM Project Management	**2,8**	**4**	**Normally**	**Always**
PM.1 Development Management	2,8	4	Normally	Always
PM.2 Communication Management	0	0	Does not Apply	Does not Apply
SU Sustainability	**2,94**	**3,2**	**Normally**	**Normally**
SU.1 Documentation	2,8	3,6	Normally	Normally
SU.2 Implementation Cost	4	4	Always	Always
SU.3 Easiness	2	2	Rarely	Rarely

translated into Aggregation of Sample Scale applying Formula 4.3 (see Table 4.5) and their outcomes are presented in Table 4.10 and Table 4.11.

Analysis

The global result of E_{ARS} is "Very Good" in the Aggregation of Sample scale (see Table 4.11) while C_{UC} is "Good" (see Table 4.10). Then E_{ARS} is better than C_{UC} and it is shown in Figure 4.4.

E_{ARS} improves C_{UC} in almost all the dimensions which is reported as follows:

Service Oriented Dimension The grades of all the experimental groups were into the range of "Normally", but these grades are different (see Table 4.9). These results were due to lack of refinement of some models.

E_{ARS} achieved in this dimension "Very Good" owing to the service models and the domain model might lack some inputs, outputs, messages, or relationships. However, C_{UC} obtained "Unacceptable" in this dimension, because it is the first time in which a service-based application (SBAs) has been design in the course.

In answer to the research question "Q.1", we report that TME encompasses the major characteristics of service-oriented computing while the previous organization process (POP) did not.

Table 4.9: Tree of goals for the Academic Request System

Goals	E_1	E_2	E_3	E_1	E_2	E_3
Global Evaluation	**3,63**	**3,5**	**3,54**	**Normally**	**Normally**	**Normally**
SO Service Oriented	**3,5**	**3,25**	**3,67**	**Normally**	**Normally**	**Normally**
SO.1 Service Description	3,5	3,25	3,67	Normally	Normally	Normally
SE Software Engineering	**3,5**	**3,5**	**3,7**	**Normally**	**Normally**	**Normally**
SE.1 Life Cycle	*3,5*	*3,5*	*3,7*	*Normally*	*Normally*	*Normally*
SE.1.1 Requirement	3	3	3,4	Normally	Normally	Normally
SE.1.2 Analysis and Design	4	4	4	Always	Always	Always
PM Project Management	**4**	**3,67**	**3,34**	**Always**	**Normally**	**Normally**
PM.1 Development Management	4	4	3,67	Always	Always	Normally
PM.2 Communication Management	4	3,34	3	Always	Normally	Normally
SU Sustainability	**3,51**	**3,58**	**3,43**	**Normally**	**Normally**	**Normally**
SU.1 Documentation	3,84	3,84	3,5	Normally	Normally	Normally
SU.2 Implementation Cost	4	4	4	Always	Always	Always
SU.3 Easiness	2,67	2,89	2,78	Normally	Normally	Normally

Table 4.10: Upshot for Control Groups

Dimensions	C_1	C_2	C_{UC}		
Global Evaluation	**2,6**	**2,65**	**2,63**	**65,75**	Good
SO Service Oriented	2	0	0	0	Unacceptable
SE Software Engineering	2,63	3,38	2,98	74,5	Good
PM Project Management	2,8	4	3,35	83,75	Very Good
SU Sustainability	2,94	3,2	3,07	76,75	Good

Table 4.11: Upshot for the Academic Request System

Dimensions	E_1	E_2	E_3	E_{ARS}		
Global Evaluation	**3,63**	**3,5**	**3,54**	**3,56**	**89**	Very Good
SO Service Oriented	3,5	3,25	3,67	3,47	86,75	Very Good
SE Software Engineering	3,5	3,5	3,7	3,57	89,25	Very Good
PM Project Management	4	3,67	3,34	3,66	91,5	Very Good
SU Sustainability	3,51	3,58	3,43	3,51	87,75	Very Good

Software Engineering Dimension. All the experimental groups obtained the mark "Normally" in the Requirement goal, because they might not have made or updated the Priority List. However, these groups fulfilled all the requirements of the Analysis and Design goal, and they achieved "Always". Consequently, the grades of all the experimental groups were into the range of "Normally" in the Life Cycle goal.

E_{ARS} improved C_{UC} owing to the former got "Very Good" at this dimension while the latter obtained "Good".

In answer to the research question "Q.2", we report that TME addresses better the life cycle of the service-based applications than the previous process.

Project Management Dimension. E_1 and E_3 achieved 'Always" in the Development Management goal, because they fulfilled all the requirements. However, E_3's grade was into the range of "Normally" owing to some models might not be updated after occurring changes.

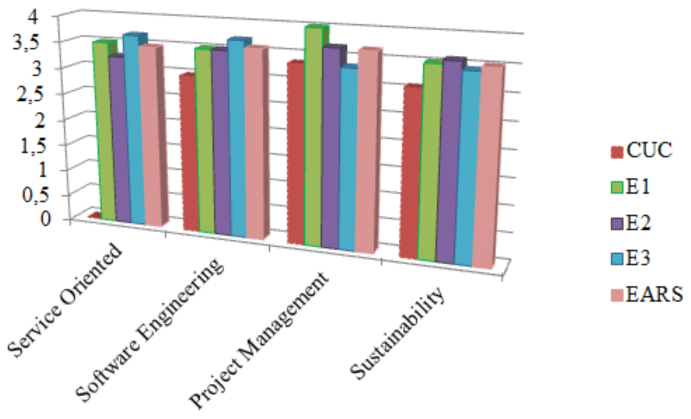


Figure 4.4: Comparison among C_{UC}, E_1, E_2, E_2 and E_{ARS}

E_1 met the “Communication Management” goal while E_2 and E_3 obtained “Normally” due to not make the meeting minutes. The control groups did not answer “Communication Management” goal, and they got “Does not Apply”.

E_{ARS} and C_{UC} achieved “Very Good” in this dimension. Having had a draw, the individual grades were considered, and the E_{ARS}'s grade was higher than C_{UC}'s grade. Thus, the response to research question “Q.3” is that TME is better in the Project Management dimension than the POP.

Sustainability Dimension. E_1, E_2, and E_3 reported that they might not have refined, and updated documents and models. For these reasons, they achieved “Normally” in the “Documentation” goal.

As far as the Easiness sub-goal is concerned, the students reported that the elaboration of models were not easy for them. In addition to this, they said that they had not studied various topics of the course by the time they carried out the design. It was observed that the major pitfalls were in the elaboration of the BPMNs, the class diagrams, and the service models. Thus, all the experimental groups had grades into the “Normally” in the “Easiness” sub-goal (see Table 4.9).

Despite of that the professors did not suggest the level of details required for the BPMNs, the students made them very detailed. Thus, they described irrelevant sub-processes as well. This is a valid argument so that TME will propose guidelines to elaborate its models.

E_{ARS} achieved in this dimension “Very Good” while C_{UC} achieved “Good”; then E_{ARS} improves C_{UC}.

In answer to the research question “Q.4”, we report that TME is better in this dimension than the POP.

Validity Threats

The main risks of validity are that the students have misunderstood of the questions and their ignorance about the software engineering concepts. On one side, the students had no experience in software development. On the other side, they might have been over-optimistic, this means that they could have filled the questionnaires only to pass the exam.

Moreover, it is the first time in which a service-oriented methodology (SOM) is used in the course, for this reason, we could not compare TME with another SOM.

4.2.2 Case Study 2: Face Recognition System

Organization

It is the same organization of case study 1 (see Section 4.2.1).

Participants

The case study was also organized as case study 1 (i.e by three groups of three students each one), but the students were not the same. However, the professor and assistants are the same of the case study 1, the profile of all the participants are the same of the those study.

Face Recognition System (FRS)

FRS aims at registering both entrance time and departure time of functionaries and professors to their labour. The registration is done through capturing their face with an artefact that compares the face with photos kept into the database and then it save the hour. This information is exchange with the Human Resource System of the university that runs in the server located in the headquarters.

Before implementing FRS, the registration uses to carry out with bar-code reader. This practice had the pitfall of anyone could have marked the assistance of another person.

The first stage of FRS will involve 350 functionaries and 1.300 professors approximately. These users are only of the branch of Asunción.

Definition of the Assessment Scope

This study has the same scope of case study 1 (see Section 4.2.1).

Data Collection

The control group is C_{UC} which have already described Section 4.2.1. The experimental groups are E_4, E_5, and E_6 which reported about their experience using TME to design FRS.

The questionnaires were sent by mail to the professor, the assistant, and the leaders of the experimental groups. After the questionnaires were completed, they were returned to the sender.

Measuring Process

First of all, the responses of all the questionnaires were included into the template which generated the results for the goal of the trees. The templates was configured according to the process defined in Section 4.1.2, and it obtained the general results for FRS that is E_{FRS}.

Moreover, the trees valued for E_4, E_5, and E_6 are shown in Table 4.12, and these results are in the Aggregation of Frequency Metric. The outcomes were translated into the Aggregation of Sample scale applying Formula 4.3, and Table 4.13 shows the translation results.

Analysis

The global result of E_{FRS} is "Very Good" in the Aggregation of Sample scale (see Table 4.13) while C_{UC} is "Good" (see Table 4.10). Then E_{FRS} is better than C_{UC} and it is shown in Figure 4.5.

Service Oriented Dimension. E_4 and E_6 did not embrace all the relationships of the services, for this reason, their grades were into "Normally" range. However, E_5 fulfilled this dimension reaching the best score "Always".

As result, E_{FRS} reached "Very Good" while C_{UC} obtained "Unacceptable".

In answer to the research question "Q.1", we report that TME encompasses the major characteristics of service-oriented computing while the POP did not.

Table 4.12: Upshot of Goals and Sub-Goals of the Face Recognition System

Goals	E_4	E_5	E_6	E_4	E_5	E_6
Global Evaluation	**3,55**	**3,48**	**3,58**	**Normally**	**Normally**	**Normally**
SO Service Oriented	**3,75**	**4**	**3,75**	**Normally**	**Always**	**Normally**
SO.1 Service Description	3,75	4	3,75	Normally	Always	Normally
SE Software Engineering	**3,7**	**3,6**	**3,7**	**Normally**	**Normally**	**Normally**
SE.1 Life Cycle	*3,7*	*3,6*	*3,7*	*Normally*	*Normally*	*Normally*
SE.1.1 Requirement	3,4	3,2	3,4	Normally	Normally	Normally
SE.1.2 Analysis and Design	4	4	4	Always	Always	Always
PM Project Management	**3,34**	**3,09**	**3,42**	**Normally**	**Normally**	**Normally**
PM.1 Development Management	3,67	3,84	3,5	Normally	Normally	Normally
PM.2 Communication Management	3	2,34	3,34	Normally	Rarely	Normally
SU Sustainability	**3,38**	**3,23**	**3,41**	**Normally**	**Normally**	**Normally**
SU.1 Documentation	3,67	3	3,34	Normally	Normally	Normally
SU.2 Implementation Cost	4	4	4	Always	Always	Always
SU.3 Easiness	2,45	2,67	3	Rarely	Normally	Normally

Table 4.13: Upshot of Dimensions of the Face Recognition System

Dimensions	E_5	E_6	E_7	E_{FRS}		
Global Evaluation	3,55	3,48	3,58	3,54	88,5	Very Good
SO Service Oriented	3,75	4	3,75	3,83	95,75	Very Good
SE Software Engineering	3,7	3,6	3,7	3,67	91,75	Very Good
PM Project Management	3,34	3,09	3,42	3,28	82	Very Good
SU Sustainability	3,38	3,23	3,41	3,35	83,75	Very Good

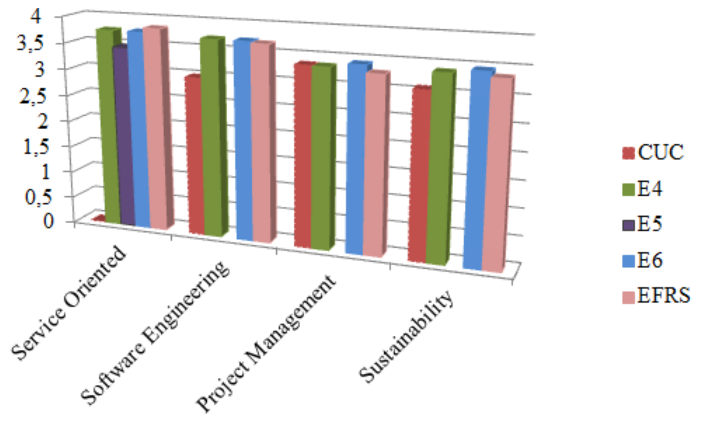


Figure 4.5: Comparison among C_{UC}, E_4, E_5, E_6 and E_{FRS}

Software Engineering Dimension. Since all the experimental groups might not have taken the requirements from users, and none of them settled the priority list with the users, they met “Normally” in the Requirement goal. In contrast, they achieved “Always” in the Analysis and Design goal. Thus, their grades were in the range of “Normally” in the Life Cycle goal.

E_{FRS} improved C_{UC} owing to the former got “Very Good” at this dimension while the latter obtained “Good”.

In answer to the research question “Q.2”, TME addresses better the life cycle of the service-based applications than the POP.

Project Management Dimension. The grades of the experimental groups were in the range of "Normally", because the schedule might not have been updated, and the meeting minutes were hardly elaborated (see Table 4.13).

The control groups did not answer the "Communication Management" goal, for this reason, they had "Does not Apply" (see Table 4.13). However, the grades of E_5 and E_6 were in the range of "Normally" and E_5 got "Rarely" in the "Communication Management".

In answer to the research question "Q.3", TME addresses better the Project Management dimension than the POP.

Sustainability Dimension. The grades of the experimental groups were into the range of "Normally" in this dimension, owing to the students might not have refined and updated the documents and models. Thus, they achieved "Normally" in the "Documentation" goal.

This case study was the first experience for the students in software design, and they were learning about UML and BPMN. For these reasons, they said that the elaboration of the models had been hard for them. Moreover, we observed the design normally included redundant details. Consequently, the grades in the Easiness goal were in the range of "Rarely" for E_4 and "Normally" for E_5 and E_6.

E_{FRS} achieved in this dimension "Very Good" while C_{UC} had "Good". This means that E_{FRS} is more sustainable than C_{UC}.

In answer to the research question "Q.4", we report that TME is more sustainable than the POP.

Validity Threats

It is the same validity of case study 1 (see Section 4.2.1).

4.2.3 Case Study 3: Information Exchange System

Organization

This case study was carried out by public agency of Paraguayan government (PIP). The PIP was set up in April 2012 and, before using TME, its development process used to be informal, according to the information collected from its quality analyst in October 2012.

Besides, it involved the Civil Registration Office, Ministry of Education, Ministry of Heath, National Police, and Ministry of Treasury.

Participants

The development of IES involved six teams, but we were able to obtain information only from one of them which was the PIP team. The PIP team was composed of a project manager, a leader, an architect, a quality analyst, and three developers. The project manager and one of the developers have a degree in computer science; the leader, the architect, the quality analyst, and one of the developers are system engineers; one developer is a student; and finally, the quality analyst is a Ph.D. student.

As far as professional experience is concerned, the project manager, the leader, the architect, the quality analyst, and one of the developer have been working in diverse projects, for more than ten years. One developer worked in a project before the IES, while for the other developer this is its first work.

Information Exchange System (IES)

The PIP used TME to manage the development of the IES, which had as its main goal the interoperability of legacy systems in public agency through the use of service-oriented computing. The operation of the mentioned IES is as follows: first, it consumes services from the providers; second, it integrates these services into a new single service; and third, it provides this new service (see Figure 3.9). It will take some years to complete the IES. Therefore, this first evaluation encompasses the period from October 2012 to March 2013.

During this period, four services were developed that exchanged citizens' data from five legacy systems that belonged to different public agencies. Each agency has a development team that is represented by a leader. The leader of the PIP is in charge of managing the communication among all leaders, and the PIP is responsible for defining the interoperability standards such as data format (see Figure 3.9), request process and to access to a service etc. The development team is in charge of services development for the owner of the legacy systems (provider or consumer). At the moment, there are three consumers, but this number is planned to increase in the future.

The first part of the IES development was financed by a sponsor who monitored the development and demanded that his own quality standards to be followed. The sponsor support lasted until March of 2013.

The IES construction was delayed several times before due to some external factors such as political reasons, lack of a law that regulates exchange of sensitive data, and staff changes. The regulating of exchange of sensitive data was only passed on January 2013, which facilitated the decision making regarding the system.

Definition of the Assessment Scope

The evaluation encompassed from the Inception Phase to the Transition Phase, over the course of six months. Table 4.14 shows the goals involved in the assessment. As a result, the assessment included

Table 4.14: Tree of Goals

SO	**Service Oriented**
SO.1	Service Description
SO.2	Service Implementation
SO.3	Exchange Standards
SE	**Software Engineering**
SE.1	Life Cycle
SE.1.1	Requirement
SE.1.2	Analysis and Design
SE.1.3	Development and Deployment
SE.2	Quality Assurance
SE.2.1	Quality Management
SE.2.2	Configuration and Change Management
SE.2.3	Test
PM	**Project Management**
PM.1	Development Management
PM.2	Communication Management
PM.3	Human Resource Management
PM.3.1	Recruitment of Human Resources
PM.3.2	Role Assignment
PM.3.3	Monitoring and Controlling Human Resource
PM.4	Procurement management
SU	**Sustainability**
SU.1	Documentation
SU.2	Implementation Cost

4 dimensions and 20 goals which involved 83 questions.

Data Collection

The interviewees where selected according to their knowledge on the whole process proposed by TME. Furthermore, it was required that they were part of the IES team of during the period from October 2012 to March 2013. Only the team leader and the quality analyst fulfilled these requirements.

The questionnaire is written in the local language (Spanish), and it takes 20 minutes approximately to be filled by the practitioners.

Having completed the questionnaires, the numeric value of each selected response is registered in the Excel template to generate all the values of the tree (see Figure 4.3).

The control questionnaire (C_{IES}) was answered by the quality analyst before applying TME. After six months of TME implementation, its performance was examined from the viewpoint of team leader (E_7) and the quality analyst (E_8) who filled in the experimental questionnaires.

Measuring Process

All the questionnaires were registered in the template, and then the trees for each of them were generated (see Section 4.1.1). Thus, the outcomes of the goals and sub-goals of the three trees of goals (i.e, 1 before TME and 2 after adopting TME) are shown in Table 4.15. Note that the results are in Aggregation of Frequency Scale, and the questions (which are measured in frequency metric) are not presented (see Table 4.4). Since there are two questionnaires (E_7 and E_8) for the experimental group, the dimensions and global evaluation (root) of their trees are combined through geometric mean (see Formula 4.2). The upshot is called E_{IES}, and it is transformed in Aggregation of Sample Scale applying Formula 4.3 (see Table 4.5).

Table 4.16 presents the outcomes of the dimensions and global evaluation of C_{IES}, E_7, E_8, and E_{IES}. C_{IES} was transformed to the Aggregation of Sample Scale to be compared with E_{IES}.

Analysis

The global evaluation for the case study is “Good” against “Unacceptable” of the organization previous process that suggests a progress in the development process of the organization. This is illustrated in Figure 4.6 which also shows the outcomes of the individual groups.

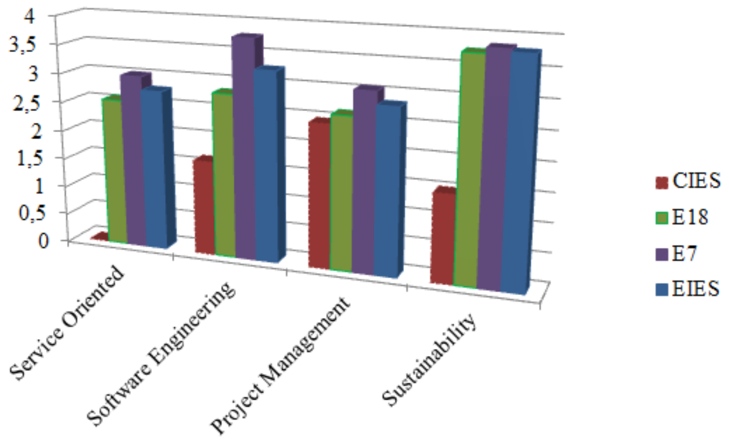


Figure 4.6: Comparison among C_{IES}, E_7, E_8, and E_{IES}

We can observe that the global evaluation for E_7 and E_8 is into “Normally” range. Note that these grades are in the Aggregation of Frequency scale (see Table 4.3).

Table 4.15: The tree of goals with the results of the three questionnaires

Goals	C_{IES}	E_7	E_8	C_{IES}	E_7	E_8
Global Evaluation	**1,87**	**2,94**	**3,42**	**Rarely**	**Normally**	**Normally**
SO Service Oriented	**0**	**2,56**	**3**	**Does not Apply**	**Normally**	**Normally**
SO.1 Service Description	0	2,67	3	Does not Apply	Normally	Normally
SO.2 Service Implementation	0	2	1	Does not Apply	Rarely	Never
SO.3 Exchange Standards	0	3	3	Does not Apply	Normally	Normally
SE Software Engineering	**1,6**	**2,82**	**3,77**	**Rarely**	**Normally**	**Normally**
SE.1 Life Cycle	*1,42*	*3*	*3,84*	*Rarely*	*Normally*	*Normally*
SE.1.1 Requirement	1,5	2,8	4	Rarely	Normally	Always
SE.1.2 Analysis and Design	1,34	3	3,5	Rarely	Normally	Normally
SE.1.3 Development and Deployment	0	3,2	4	Does not Apply	Normally	Normally
SE.2 Quality Assurance	*1,8*	*2,64*	*3,7*	*Rarely*	*Normally*	*Normally*
SE.2.1 Quality Management	1	2,8	3,4	Never	Normally	Normally
SE.2.2 Configuration and Change Management	2	2,6	4	Rarely	Normally	Always
SE.2.3 Test	2,5	2,5	0	Normally	Normally	Does not Apply
PM Project Management	**2,46**	**2,62**	**3,05**	**Rarely**	**Normally**	**Normally**
PM.1 Development Management	3,5	3,86	3,58	Normally	Normally	Normally
PM.2 Communication Management	1	1,25	2	Never	Rarely	Rarely
PM.3 Human Resource Management	*3,34*	*3,17*	*3,34*	*Rarely*	*Normally*	*Normally*
PM.3.1 Recruitment of Human Resources	4	4	4	Always	Always	Always
PM.3.2 Role Assignation	2,5	2,5	2,5	Normally	Normally	Normally
PM.3.3 Monitoring and Controlling Human Resource	3,5	3	3,5	Normally	Normally	Normally
PM.4 Procurement Management	2	2,17	3,25	Rarely	Rarely	Normally
SU Sustainability	**1,5**	**3,75**	**3,84**	**Rarely**	**Normally**	**Normally**
SU.1 Documentation	1,5	3,5	3,67	Rarely	Normally	Normally
SU.2 Implementation Cost	0	3	4	Does not Apply	Normally	Always

Table 4.16: Upshot of Dimensions

Dimensions	C_{IES}	E_7	E_8	E_{IES}	C_{IES}		E_{IES}	
Global Evaluation	**1,87**	**2,94**	**3,42**	**3,17**	**46,33**	**Acceptable**	**79,25**	**Good**
SO Service Oriented	0	2,56	3	2,77	0	Unacceptable	69,25	Good
SE Software Engineering	1,6	2,82	3,77	3,26	40	Unacceptable	81,5	Very Good
PM Project Management	2,46	2,62	3,05	3,36	60	Good	84	Very Good
SU Sustainability	1,5	3,75	3,84	3,79	37,5	Unacceptable	93,5	Very Good

The outcomes of the dimension reveals the following:

Service Oriented Dimension. The upshot of the combination of the two experimental groups (E_{IES}) is "Good", this means that more than 60% of the time, the suggestions of TME have been observed (see Table 4.16).

Even though TME suggests signing the Service Level Agreement (SLA), the agreement was not formalized. For this reason, The grade in the Service Description goal was "Normally".

The quality analyst did not give an answer on the "Service Implementation", because she did not participate in the development. This is the weakest goal of this dimension due to various suggestions of TME was not taken into account, such as the service repository (UDDI) creation.

Moreover, the grade in the Exchange Standards goal was "Normally" due to not have implemented techniques for the concepts unification.

C_{IES} has obtained "Does Not Apply" in this dimension because it is the first time in which a service-based application (SBAs) has been used in the PIP.

We can observe that TME has improved the POP in response to the research question "Q.1".

Software Engineering Dimension. There are different responses in Requirement and Configuration, and Change Management due to practices and documents proposed to them; but they have not always been considered by the developers.

The quality analyst suggested to follow certain standards during the development, but it was not her responsibility to guarantee that the developers really followed the suggested standards.

Testing has not always been carried out by other people besides developers and there was not a responsible person for accepting the development before deploying. Besides, verification and validation practices were not suggested at that moment.

E_{IES}'s grade was into "Very Good" range in this dimension, which means that more than the 80% of the time the suggestions of TME in the "Software Engineering" area have been observed. This result suggests that TME has improved the POP, considering the research question "Q.2" (see Table 4.16).

Project Management Dimension. Various processes and documents were not followed and completed because they were unknown by the majority of the staff due to a fault in Communication Management. This situation is reflected in the very low score in "Communication Management", and affected the Role Assignation because some members of the team were not notified about their responsibilities.

Part of the IES development was outsourced, and the contract with the provider was signed before applying TME. For this reason, several of TME's suggestions were not been considered.

Sustainability Dimension. Even though the models were not easily understandable to some of the staff, the documentation was usually updated.

The PIP determined that all the tools used for developing IES should be open source. These tools include database, programming language, modelling language, project management software, version control software, etc. Moreover, TME includes standard notation such as UML and BPMN.

However, the lack of documentation of the applications developed at the PIP before applying TME has hindered their maintenance.

Since C_{IES} obtained the "Unacceptable" grade, when comparing it to the "Very Good" grade obtained by E_{IES}, we can consider that TME has become more sustainable than the POP in response to research question "Q.4" (see Table 4.16).

Validity Threats

The IES mentioned in this study is the first service-based application implemented in the public sector in Paraguay. For this reason, we could not compare TME with another development process for service-based application in the same context.

Since the questionnaires are the tool used to measure the achievement of the goals, their reliability affects the validity of the evaluation. For this reason, they have been elaborated to embrace all the aspects related with the goals and their questions strive to be accurate and easy to understand. It is also important to note that the validity is constrained by the skills and perception of the interviewees.

Another limitation is that only two people were interviewed. Even though these two persons are considered proficient professionals, and they have been in charge of important projects in public and private sector in Paraguay.

4.2.4 Case Study 4: Financial System

Organization

Software Factory (SF) is a company developing of software for different public agencies. It had 65 people working in software construction, and 20 people in other areas. Moreover, it can have various independent projects executing in parallel. Each project is normally assigned to a team whose size depends on different factors such as software size, schedule, and budget.

The organization had not defined a development process to be followed by all the development teams, for this reason, it started to experiment TME in two projects: the Financial System and the Healthcare System.

Participants

The team was composed of five people, one of them was the team leader.

As far as the profile of the staff is concerned, the team leader is Ph.D. while the other members just graduated, and this study was either the first or second project in which they worked.

Financial System (FIS)

It automates financial procedures of a public bank that focuses on giving credits with low interests to families with low incomes to buy their dwelling. The system has been implemented with a tool called Genexus for decision of the client. The team performed their labour in the client's office, and they arranged their meetings twice a week. Further, the meetings with the client have been carried out once a week.

Moreover, all the banks of the country are regulated by the Central Bank of Paraguay (BCP) which compels to them to observe the COBIT [122]. Thus, the COBIT was taken into account in the construction of FIS which has been periodically audited. The success in the first audit of FIS motivated the use of TME in another project which had started two years before, and it had problems of organization; the project was the **Healthcare System (HCS)**.

Definition of the Assessment Scope

The assessment involves the Inception, the Elaboration, and the Transition phases. Moreover, the tree of goals involved in our analysis is shown in Table 4.17 where we can observe that This tree excludes the "Service Oriented Dimension" and the "Implementation Cost Goal" were excluded from the assessment.

Thus, the assessment included 3 dimensions and 13 goals which involved 57 questions.

Data Collection

The data has been gathered during an interview using a questionnaire.

The researcher asked the questions to interviewees for filling in the questionnaire. The interviewees were the quality analyst of SF, and the team leader of Financial System (E_{FIS}). The team leader answered about FIS, and its interview is part of the experimental group.

Table 4.17: Tree of Goals for the software factory

SE	**Software Engineering**
SE.1	Life Cycle
SE.1.1	Requirement
SE.1.2	Analysis and Design
SE.1.3	Development and Deployment
SE.2	Quality Assurance
SE.2.1	Quality Management
SE.2.2	Configuration and Change Management
SE.2.3	Test
PM	**Project Management**
PM.1	Development Management
PM.2	Communication Management
PM.3	Human Resource Management
SU	**Sustainability**
SU.1	Documentation
SU.2	Easiness

Additionally, the quality analyst reported how had been carried out the development before applying TME; this is used as the control group (C_{SF}).

Measuring Process

Both control group (C_{SF}) and experimental group (E_{FIS}) are composed of only one person.

Moreover, the questionnaires were registered in the template, and then the trees for each of them were generated (see Section 4.1.1). Thus, the outcomes of the goals and sub-goals of the three trees of goals are shown in Table 4.18 and they are in the Aggregation of Frequency Scale.

Table 4.18: Upshot of Goals and Sub-Goals of the Financial System

Goals	C_{SF}	E_{FIS}	C_{SF}	E_{FIS}
Global Evaluation	**2,6**	**3,49**	**Normally**	**Normally**
SE Software Engineering	**2,59**	**3,75**	**Normally**	**Normally**
SE.1 Life Cycle	*2,34*	*3,49*	*Rarely*	*Normally*
SE.1.1 Requirement	2,75	4	Normally	Always
SE.1.2 Analysis and Design	1,75	3,25	Rarely	Normally
SE.1.3 Development	2,5	3,2	Normally	Normally
SE.2 Quality Assurance	*2,83*	*4*	*Normally*	*Always*
SE.2.1 Quality Management	2,88	4	Normally	Always
SE.2.2 Configuration and Change Management	2,6	4	Normally	Always
SE.2.3 Verification and Validation	3	4	Normally	Always
PM Project Management	**3,31**	**3,73**	**Normally**	**Normally**
PM.1 Development Management	3	3,72	Normally	Normally
PM.2 Communication Management	3,25	3,8	Normally	Normally
PM.3 Human Resource Management	3,67	3,67	Normally	Normally
SU Sustainability	**1**	**2,97**	**Never**	**Normally**
SU.1 Documentation	1	3,4	Never	Normally
SU.2 Easiness	0	3,34	Does not Apply	Normally

The outcomes were transformed to the Aggregation of Sample Scale applying Formula 4.3, and they are shown in Table 4.19.

Table 4.19: Upshot of Dimensions of the Financial System

Dimensions	C_{SF}	E_{FIS}	C_{SF}		E_{FIS}	
Global Evaluation	**2,64**	**3,49**	66	Good	87,25	Very Good
SE Software Engineering	2,59	3,75	3,71	Good	93,75	Very Good
PM Project Management	3,31	3,73	3,74	Very Good	93,25	Very Good
SU Sustainability	1	3,49	3,51	Unacceptable	87,25	Very Good

Analysis

The global result of E_{FIS} is "Very Good" in the Aggregation of Sample scale while C_{SF} is "Good" (see Table 4.19). Then E_{FS} is better than C_{SF} and it is shown in Figure 4.7.

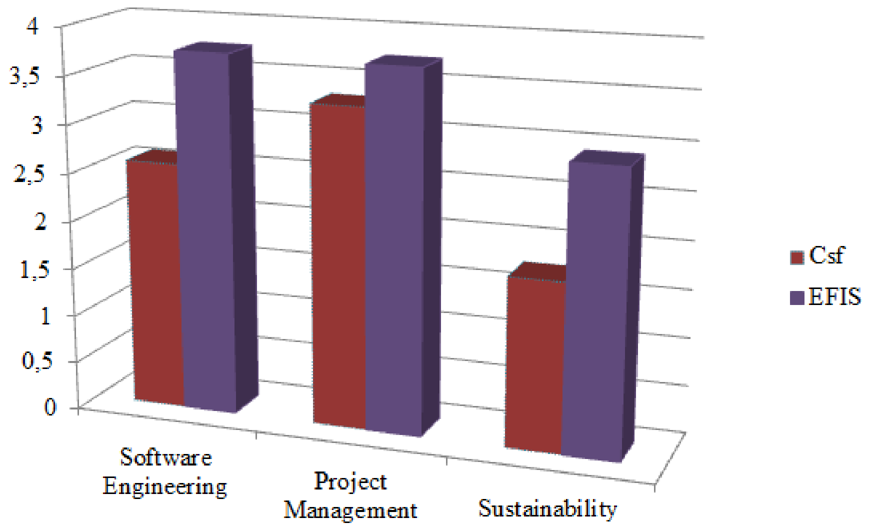


Figure 4.7: Comparison among C_{SF} and E_{FIS}

The analysis of the results of the dimensions informs the following:

Software Engineering Dimension. E_{FIS} fulfilled the needs of the Requirement sub-goal while C_{SF} failed in some aspects. The grade of C_{SF} was in the range of "Rarely" in the Analysis and Design sub-goal while E_{FIS} reached "Normally" in this sub-goal. These outcomes were due to the lack of some models.

All the suggestions of TME were met for E_{HCS} in the Development sub-goal while

E_{FIS} might not have fulfilled the Development sub-goal, as a result, it had "Normally" grade in this sub-goal. Moreover, C_{SF} reported that the previous process of the organization did not normally consider the good practices of programming; then its grade was in the range of "Rarely" in the Development sub-goal. Thus, E_{FIS} addressed better the Life Cycle goal than C_{SF}.

E_{FIS} fulfilled the Quality Assurance goal reaching "Always" (see Table 4.18). Notwithstanding, the grade of C_{SF} was in the range of "Normally" the Quality Assurance goal. Among various cause of these outcomes, there were normally non-observance of the standard of code and interface; it might not have been tracked the changes in the project's assets; and the software deliverable was not always validated.

E_{FIS} achieved in this dimension "Very Good" while C_{SF} had "Good" in this dimension. For this reason, the response to the research question "Q.2" is that TME is better than the POP in the Software Engineering Dimension.

Project Management Dimension. The grades of both groups were in the range of "Normally" in the Development sub-goal owing to the teams might not have updated the chart after changes, and they might not have made the iterations plan. However, C_{SF} reported that these shortcomings were more frequent in the previous organization process.

All the groups obtained the same grade the Human Resources Management sub-goal, but for different reasons. For instance, C_{SF} might not have controlled the staff while E_{FIS} might not have notified its human resources about their responsibilities.

The grades of E_{FIS} and C_{FS} are in the range of "Very Good" in this dimension, but the individual grades showed that E_{FIS}'s grade was higher than C_{FS}'s grade. Therefore, the response to research question "Q.3" is that TME is better in the Project Management dimension than the POP.

Sustainability Dimension. C_{SF} reported that the previous organization process did not propose the elaboration of neither documents or models. As a result, the C_{SF}'s grade was "1" in the Documentation goal, and the Easiness goal could not be evaluated (see Table 4.18).

The grades of E_{FIS} was in the range of "Normally" owing to the team might not update the requirement documents after changes in them. In addition to this, E_{FIS} reported that the FIS's architecture was not normally refined and updated.

E_{FIS} reached "Very Good" in this dimension while C_{SF} had "Unacceptable" (see Table **??**). As a consequence, E_{FIS} is more sustainable than C_{SF}.

In answer to the research question "Q.4", we report that TME is better than the POP in this dimension.

Validity Threats

It is low the risk that the questions were misunderstood due to the interview was held by the researcher.

The critical factor was that the questions had encompassed all the aspects to be included in the evaluation.

Regarding to a small number of interviewees, their wide experience managing software projects make valuable their opinion.

4.2.5 Case Study 5: Healthcare System

Organization

It is the same organization of case study 4 (see Section 4.2.4).

Participants

This study involved various teams and one quality analyst; and each of them is in charge of one module of the system. The development of this system involved approximately 33 people. The teams is composed of 15 students, 8 system engineers, 5 bachelors in computers science, 3 masters, and 2 PHDs. Among the masters is the quality analyst.

Healthcare System (HCS)

HCS automatizes the procedures of a Public Hospital (Hospital de Clinicas) that is one of the biggest public hospital of the capital. The hospital is a dependency of the Medicine School of UNA[2], and the

[2]http://www.med.una.py/Historia.html

medical students have their practical courses there.

Definition of the Assessment Scope

This study has the same scope of case study 4 (see Section 4.2.4).

Data Collection

The data was gathered during an interview with the quality analyst of HCS. This interview was taken as experimental group (E_{HCS}).

As far as the control group is concerned, we used the control group C_{SF} of the case study 4, because both studies belong of the same organization.

Measuring Process

The questionnaire was measured according to the process described in Section 4.1.2. The grades are in the Aggregation of Frequency Scale and are shown in Table 4.20. This outcomes were translated into the Aggregation of Sample Scale applying Formula 4.3, and they are shown in Table 4.21.

Table 4.20: Upshot of Goals and Sub-Goals of the Healthcare System

Goals	C_{SF}	E_{HCS}	C_{SF}	E_{HCS}
Global Evaluation	**2,64**	**3,54**	**Normally**	**Normally**
SE Software Engineering	**2,59**	**3,67**	**Normally**	**Normally**
SE.1 Life Cycle	*2,34*	*3,5*	*Rarely*	*Normally*
SE.1.1 Requirement	2,75	4	Normally	Always
SE.1.2 Analysis and Design	1,75	2,5	Rarely	Rarely
SE.1.3 Development	2,5	4	Normally	Always
SE.2 Quality Assurance	*2,83*	*3,84*	*Normally*	*Normally*
SE.2.1 Quality Management	2,88	3,88	Normally	Normally
SE.2.2 Configuration and Change Management	2,6	3,8	Normally	Normally
SE.2.3 Verification and Validation	3	3,84	Normally	Normally
PM Project Management	**3,31**	**3,75**	**Normally**	**Normally**
PM.1 Development Management	3	3,58	Normally	Normally
PM.2 Communication Management	3,25	4	Normally	Always
PM.3 Human Resource Management	3,67	3,67	Normally	Normally
SU Sustainability	**1**	**3,2**	**Never**	**Normally**
SU.1 Documentation	1	3,4	Never	Normally
SU.2 Easiness	0	3	Does not Apply	Normally

Table 4.21: Upshot of Dimensions of the Healthcare System

Dimensions	C_{SF}	E_{HCS}	C_{SF}		E_{HCS}	
Global Evaluation	**2,64**	**3,54**	66	Good	88,5	Very Good
SE Software Engineering	2,59	3,67	64,75	Good	91,75	Very Good
PM Project Management	3,31	3,75	93,75	Very Good	93,5	Very Good
SU Sustainability	1	3,54	25	Unacceptable	88,5	Very Good

Analysis

The global result of E_{HCS} is "Very Good" in the Aggregation of Sample scale while C_{SF} is "Good" (see Table 4.21), then E_{HCS} is better than C_{SF}, as it is shown in Figure 4.8.

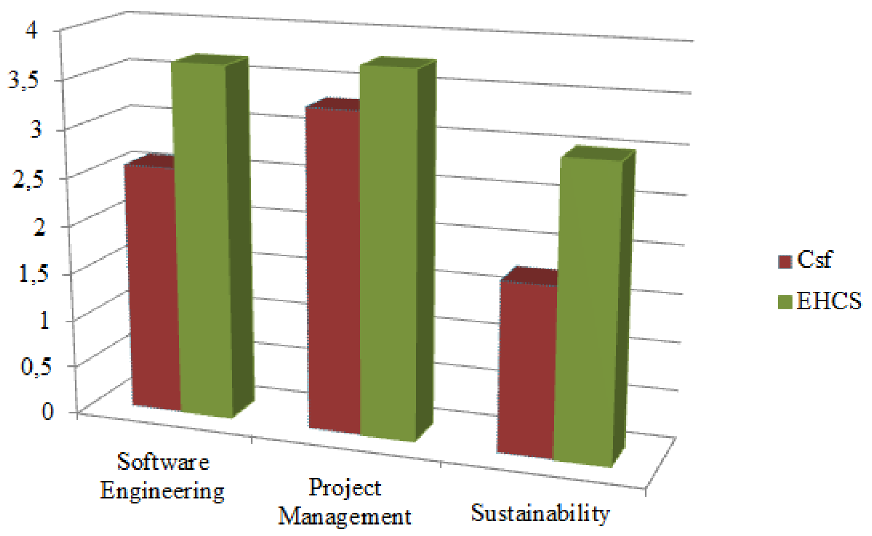


Figure 4.8: Comparison among C_{SF} and E_{HCS}

The analysis of the results of the dimensions informs the following:

Software Engineering Dimension. E_{HCS} fulfilled the needs of the Requirement sub-goal while C_{SF} missed some aspects. The grades of C_{SF} and E_{HCS} were into the range of "Rarely" in the Analysis and Design sub-goal.

All the suggestions of TME were met for E_{HCS} in the Development sub-goal. Conversely, C_{SF} reported that the previous organization process did not consider the good practices of programming; then its grade was in the range of "Rarely" in the Development sub-goal. Thus, E_{HCS} managed better the Life Cycle goal than C_{SF}.

The grades of E_{HCS} and C_{SF} were in the range of "Normally" the Quality Assurance goal, but E_{HCS}'s grade is higher than C_{SF}'s.

Thus, the response to the research question "Q.2" is that TME is better than the POP in the Software Engineering Dimension.

Project Management Dimension. The grades of E_{HCS} and C_{SF} were in the range of "Normally" in the Development sub-goal owing to the teams might not have updated the chart after changes, and they might not have made the iterations plan. However, C_{SF} reported that these shortcomings were more frequent in the previous process, and then its grade is lower than E_{HCS}'s grade.

E_{HCS} met all the needs of the Communication Management sub-goal, thus, it got "Always". Meanwhile, The grades of the other groups were in the range of "Normally" due to the meeting minutes were not always elaborated.

All the groups obtained the same grade in the Human Resources Management sub-goal that is in the range of "Normally" owing to lack of tracking and controlling the staff.

Both groups achieved "Very Good" in this dimension, but the grade of E_{HCS} was higher than the grade C_{UC}. Therefore, the response to research question "Q.3" is that TME is better in the Project Management dimension than the POP.

Sustainability Dimension. C_{FS} reported that the previous process did not propose the elaboration of neither documents or models. As a result, its grade was "1" in the Documentation goal, and the Easiness goal could not be evaluated (see Table 4.20). However, the grade of E_{HCS} was in the range of "Normally" owing to the team might not update the requirement documents after changes in them.

E_{HCS} reached “Very Good” in this dimension while C_{SF} had “Unacceptable” (see Table 4.21). As a consequence, in answer to the research question “Q.4”, we report that TME is more sustainable than the POP.

Validity Threats

It is the same validity of case study 4 (see Section 4.2.4).

4.3 Analysis Phase

To verify that TME improves current practices of the different organizations, we applied meta-analysis (see Section 4.1.3). The meta-analysis technique implemented was the vote-counting method that calculates an improvement magnitude calls it effect-size.

The effect-size interpretation is given in Table 4.22 [118]. We used the outcomes in the Aggregation of Frequency Scale to compute effect-size, vote, and probability of effect-size, occurrence for each case study.

Table 4.22: Interpretation of Effect Size

Effect Size	**Effect Size Value**	**Percentile Standing**	**Per cent of non-overlap**
LARGE	0.8	79	47.4%
	0.7	76	43.0%
	0.6	73	38.2%
MEDIUM	0.5	69	33.0%
	0.4	66	27.4%
	0.3	62	21.3%
SMALL	0.2	58	14.7%
	0.1	54	7.7%
	0.0	50	0%

The results are shown in Table 4.23 where we can observe that all the case studies got positive votes (see Section 4.1.3 for the interpretation of the *Vote* values). This means that the experimental groups, which represent TME, got better results than control groups, which represent POP.

Table 4.23: Case Studies

	Groups		
Case Studies	**Experimental**	**Control**	**Vote**
ARS	3,56	2,63	1
FRS	3,54	2,63	1
IES	3,17	1,4	1
FIS	3,47	2,55	1
HCS	3,49	2,55	1

Since the case studies that belong to the same organization are compared against the same control group, we computed the standard deviation among them from global results of the experimental groups, and then we calculated the effect-size. Hence, the experimental group for the case studies of the university is UC while for the software factory is SF.

Table 4.24 shows that the effect-size of UC is “0,4” that corresponds to “MEDIUM” effect and its probability is 0,09. It indicates that the experimental groups are at the 66th percentile of the control groups, and the per cent of non-overlap is 27,4%. For IES the effect-size is “0,36” that is “MEDIUM”

Table 4.24: Meta-Analysis of Case Studies

	Groups			
Case Studies	**Experimental**	**Control**	**Effect Size**	**Effect Size Probability**
UC	3,56	3,55	0,4	0,09
IES	3,17	3,17	0,36	0,15
SF	3,47	3,48	0,36	0,15

effect (Table 4.22). This result means that the experimental groups are at the 62th percentile of the control group, and the per cent of non-overlap between both groups is 21,3%.

Moreover, the effect-size of SF is “0,36” that is “MEDIUM”. This means that the experimental groups are at the 62th percentile of the control group, and the per cent of non-overlap between both groups is 21,3%.

The major effect-size was obtained by UC, but its probability of occurrence is the lowest. However, the effect-size with the highest probability of occurrence is “0,4” that is “MEDIUM”.

Therefore, the most likely effect is “MEDIUM” that is an optimistic improvement to the POP.

4.4 Chapter summary

We have examined the performance of TME in five case studies which are the Academic Request System (ARS), the Face Recognition System (FRS), the Information Exchange System (IES), the Financial System (FIS), and the Healthcare System (HCS).

ARS and FRS were carried out in an academic context in order to refine the proposal and infer its simplicity owing it can be used by novice practitioners. While, IES, FIS and HCS were implemented in public agencies to validate TME in real e-government application domain.

The data of the studies where collected using questionnaires which were completed by both control and experimental groups (see Appendix A). The control group informed about the POP while the experimental reported about TME. Only one person was part of the control group of IES, FIS, and HCS. However, the control group of ARS and FRS had two people.

The population of the experimental groups was different. Thus, IES’s group had two people. FIS and HCS had only one person. Moreover, ARS and FRS had three experimental groups each one, and each group was composed of three students.

The control questionnaire “C_{IES}” was only compared against the experimental groups of IES. However, the control questionnaires “C_{UC}” and “C_{SF}” were used each one to contrast the experimental groups of two case studies. Thus, C_{UC} were put together with the experimental groups of both ARS and FRS while C_{SF} was done it with the experimental groups of both HCS and FIS.

Table 4.25 shows that the questionnaires can be different for each case study. Moreover, we can notice that the quantity of interviewees, questionnaires involved, dimensions, goals and questions as well as the life cycle covered vary according to the organization in which the case studies were carried out. The academic case studies included more interviewees (18 students, the professor, and one assistant), but their scope is more reduced (i.e less phases, goals, and questions). In contrast, the case studies of the public agencies encompass from the Inception Phase to the Transition Phase.

Table 4.25: Questionnaires Description

					Number			
Organization	**Sector**	**Case Study**	**Questionnaire**	**Phases Covered**	**Interviewee**	**Dimension**	**Goal**	**Question**
PIP	Public	IES	C_{IES}, E_1 , E_2	• Inception • Elaboration • Construction • Transition	2	4	20	83
Catholic University	Private	ARS	C_{UC}, E_3, E_4 , E_5	• Inception • Elaboration	10	4	13	40
		FRS	C_{UC}, E_6, E_7, E_8	• Inception • Elaboration	9	4	13	40
Software Factory	Public	HCS	C_{SF}, E_{HCS}	• Inception • Elaboration • Construction • Transition	2	3	11	57
		FIS	C_{SF}, E_{FIS}	• Inception • Elaboration • Construction • Transition	2	3	11	57

We observed that FIS and HCS managed better the Software Engineering and Project Management dimensions than the other studies (see Figure 4.9), but they did not include the service oriented dimension. Moreover, they were the less sustainable, while the most sustainable was IES.

FRS addressed better the service oriented dimension than the others. However, none of the POPs were service-oriented.

Table 4.26 summarizes the outcomes of the studies. We can observe that TME improved all the POPs. Therefore, the grades of the POPs are between "Acceptable" and "Good" while the grades of TME are between "Good" and "Very Good".

Despite the actual results are more focused on the process and do not encompass the maintenance and retirement phases as well as the quality of the final system obtained by the application of TME, the improvements obtained compared with the previous organization process are encouraging.

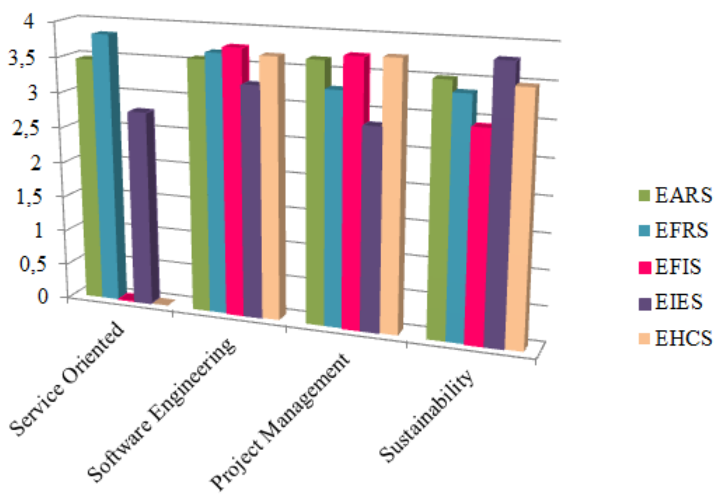


Figure 4.9: Comparison among the case studies

Table 4.26: Summary of the Case Studies

Case Studies	**Previous Processes**		**Tape Mbo'e**	
ARS	65,75	Good	89	Very Good
FRS	65,75	Good	88,5	Very Good
IES	46,33	Acceptable	79,25	Good
FIS	66	Good	87,25	Very Good
HCS	66	Good	88,5	Very Good

Chapter 5

Contributions, Conclusions and Future Research

In any chapter, we presented the most important considerations about the work done. What follows is a summary of the overall contributions and conclusions of this thesis, as well as some proposals for future research.

5.1 Contributions

At the end of this thesis, we consider that we have fulfilled the proposed objectives. Thus, our major contributions to the progress of the design and construction of e-government application in developing countries are outlined as follows:

- A critical analysis of the state of the art of the Agile Methods (AMs) with regarding life cycle coverage, project management, and sustainability was presented in Chapter 2. This has helped to identify the features that have to be included in an AM in order to be able to support the development of e-government service-based applications in developing countries.
- A detailed analysis of the Service-Oriented Methods (SOM) from the viewpoints of SOC characteristics, life cycle coverage, project management, and sustainability. This analysis suggested the SOC characteristics to be included in our proposal as well as the limitations of the analysed SOMs. It is described in Chapter 2.
- The adaptation of "OpenUP", an agile method, to support the development and maintenance of SBAs, considering the project management and the quality assurance dimensions. As a result, we have proposed Tape Mbo'e that is described in Chapter 3.
- The application of TME in five case studies that involved academic and public organization in Paraguay. These studies are described in Chapter 4.
- A detailed description of the process followed to assess TME, and its outcomes and conclusions were included in Chapter 4.
- Moreover, a relevant contribution is that TME's evaluation was the first application of the rigorous experimental evaluation in public sector in Paraguay.

Some of the results of this thesis have been published:

- Grau, I. and Travassos, G. and Cernuzzi, L. and Villafiorita, A. Tape Mbo'e: A First Experimental Assessment. CLEI Electronic Journal, Extended version of ESELAW 2014, accepted paper, 18, NUMBER X, pages 25, 2015.

- Grau, I. and Travassos, G. and Cernuzzi, L. and Villafiorita, A. Tape Mbo'e: A First Experimental Assessment. 11th Workshop on Experimental Software Engineering (ESELAW), Pucón - Chile. April, 2014.
- Grau, Ilse. Tape Mbo'e (TME). In Proc. of the 16th Iberoamerican Conference on Software Engineering (CIBSE). Montevideo-Uruguay, April, 2013.
- Grau, Ilse and Cernuzzi, Luca and Villafiorita, Adolfo. Analysing Service Oriented methods for Sustainable Applications. In Proc. of the 32nd International Conference of the Chilean Computer Science Society (SCCC). Temuco-Chile, November, 2013.
- Grau, Ilse and Cernuzzi, Luca and Villafiorita, Adolfo. Tape Mbo'e (TME). In Proc. of the 32nd International Conference of the Chilean Computer Science Society (SCCC). Temuco-Chile, November, 2013.

Finally, we are working on others papers focusing on more experimental results.

5.2 Conclusion

This thesis proposes a software engineering method to develop and maintain service-based e-government applications which can be implemented in developing countries.

Thus, at the beginning of the research we have considered two aspects which are the availability of software engineering methods and the socio-cultural conditions of a specific developing country like as Paraguay. The context of developing countries introduces additional constraints, such as the methods have to: produce fast deliveries; maintain development cost low; and, consider from the beginning the aspects related to long-term sustainability of the solutions.

As regards to sustainability, Agile Methods seemed to be the most convenient owing to the ability of those methods to adapt fast in changing environments; they focus on customer satisfaction, and provide early and continuous deliveries. However, they do not encompass the technical aspects of the Service-Oriented Computing (SOC), and they offer little or no documentation that makes difficult software maintenance.

The technical aspects are tackled by the Service Oriented Methods (SOMs), but their weaknesses are the project management and the sustainability. As far as project management is concerned, SOMs do not deal with communication, human resources, and providers. Moreover, there is little information about SOMs, and among them only Service Oriented Modelling and Architecture [13] and Quasar Enterprise Method [14, 15] have been validated in real cases; both aspects affect the sustainability.

Thus, we have analysed these kinds of methods under the perspective of technical-methodological dimension (i.e service-oriented and life cycle coverage), the project management and the sustainability using an evaluation framework. The analysis has included as representatives of AMs to Scrum [28], [29], Extreme Programming [30, 31], Open Unified Process[1], Feature Driven Development [32], and Dynamic Systems Development Method [32]. The SOMs have been represented by Service Oriented Modelling and Architecture [13], the Service Oriented Architecture Framework [37], the Service-Oriented Design and Development Methodology [38], the Quasar Enterprise Method [14, 15], the proposals of Demchak et al. [78], and Lamparter et al. [96]. The analysis has concluded that AMs are better than SOMs in the aspects related with project management and sustainability; and, among them, Open Unified Process (OpenUP) is the closest to achieve our goals. As a result, OpenUP was chosen to be the baseline of our approach which has been called Tape Mbo'e (TME).

Furthermore, TME is a service oriented method that modified the life cycle and the disciplines of OpenUP. The former covers the maintenance and retirement of software while the latter includes SOC

[1] $http://en.wikipedia.org/wiki/OpenUP$

characteristics, the quality assurance, and improvements to the project management. These changes have affected the definition of roles and work products as well. Consequently, TME introduces new roles, gives up some existent roles, and includes new work-products. The changes included in TME could reduce its agility, but they were needed by the type of projects in which we are interested.

TME has been evaluated through its application in case studies. The use of multiple case studies supports findings credibility [123, 124]. The assessment has been directed by an evaluation framework and has involved five case studies of three different organizations.

The framework is composed of three phases: Planning, Execution, and Analysis. The Planning Phase describes how to carry out the assessment, and it includes goals, measures, and scales. The Execution Phase depicts each case study with its measurements, and the analysis of its results. The Analysis Phase focuses on the meta-analysis applying the Vote-Counting Method [118, 119] to compare the outcomes of the case studies and obtain a conclusion.

The first two case studies have been carried out by students in university, and they have focused on validating the SOC characteristics, and TME's easiness to be applied by novices. The first evaluation of TME in e-government environment was the Information Exchange System (IES) that is in charge of interchanging citizens' data among the legacy systems of the whole public sector of Paraguayan government [14]. This experience has represented the first application of rigorous experimental evaluation in the public sector in Paraguay. The outcomes of this case study have been supported by two additional studies. Such studies have been accomplished by an organization that has outsourced the development of systems for two public agencies. Besides, they have focused on software engineering and project management aspects.

We have observed in all the case studies that TME has improved the previous organization process. According to vote-counting outcomes the improvement's size can be between medium and large that represent a step towards the quality of development process of the organizations.

Despite the actual results are more focused on the process and do not encompass the maintenance and retirement phases as well as the quality of the final system obtained by the application of TME, the improvements obtained compared with the previous organization process are encouraging.

5.3 Future Research

In spite of the advantages pointed out above, several other issues remain as interesting challenges for researchers and practitioners devoted to design and construction of e-government application in developing countries.

Thus, some aspects of the TME method could be improved. One of these improvements is related to the provision of techniques to identify and select potential services among either existing or new functionalities.

Another open issue is related with the services repository, the method should outline how the storage, search, and publication of services will be managed.

As far as re-usability of functionalities is concerned, TME should be able to identify the advantages or disadvantages of re-using existing assets. Besides, it should suggest mechanisms for reusing.

In addition to this, an interesting improvement is the inclusion of metrics to measure the size and/or complexity of development projects in order to propose configurations of TME according to these characteristics. For instance, one configuration for small projects and another for big one.

As regards evaluation, the final applications produced adopting TME should be assessed, as well as the behaviour of TME during the Operation Phase.

Finally, we suggest the definition of TME configurations to meet the requirements of some well-known quality standards, such as CMMI.

Bibliography

[1] V. de Castro, E. Marcos, and B. Vela. Representing WSDL with extended UML. *Revista Colombiana de Computación - RCC*, 2004.

[2] R. Klischewski. Architectures for Tinkering? Contextual Strategies towards Interoperability in E-government. *JTAER*, 6(1):26–42, 2011.

[3] S. Ochoa, P. Rossel, and C. Bastarrica. A software architecture to support digital document interchange for the Chilean government. In *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, pages 2223–2227, 2006.

[4] P. Shvaiko, A. Villafiorita, A. Zorer, L. Chemane, T. Fumo, and J. Hinkkanen. eGIF4M: eGovernment Interoperability Framework for Mozambique. In *Proceedings of the 8th International Conference on Electronic Government*, EGOV '09, pages 328–340, Berlin, Heidelberg, 2009. Springer-Verlag.

[5] I. Grau, G. Travassos, L. Cernuzzi, and A. Villafiorita. Tape mbo'e: A first experimental assessment. *CLEI Electronic Journal*, 18(1):XX, 2015. Extended version of ESELAW 2014; accepted paper.

[6] S. Roy and M. Debnath. Designing SOA based e-governance system using eXtreme programming methodology for developing countries. In *2010 2nd International Conference on Software Technology and Engineering (ICSTE)*, volume 2, pages 277–282, Oct. 2010.

[7] B. Eshete, A. Mattioli, A. Villafiorita, and K. Weldemariam. ICT for Good: Opportunities, Challenges and the Way Forward. In *Fourth International Conference on Digital Society, 2010. ICDS '10.*, pages 14–19, February 2010.

[8] R. Heeks. Information systems and developing countries: Failure, success and local improvisations. *Inf. Soc.*, 18(2), 2002.

[9] L. Cernuzzi, M. González, M. Ronchetti, A. Villafiorita, and K. Weldemariam. *Global Strategy and Practice of e-Governance: Examples from Around the World*, chapter Experiences in e-Governance from an ICT4G Perspective: Case Studies and Lesson Learned. IGI Global, 2011.

[10] H. Hajjdiab and A.S. Taleb. Agile adoption experience: A case study in the U.A.E. In *2011 IEEE 2nd International Conference on Software Engineering and Service Science (ICSESS)*, pages 31–34, July 2011.

[11] J. Stafford, K. Erdil, E. Finn, K. Keating, J. Meattle, S. Park, and D. Yoon. Software maintenance as part of the software life cycle. December 2003.

[12] S. Black, P. Boca, J. Bowen, J. Gorman, and M. Hinchey. Formal versus Agile: Survival of the fittest. *Computer*, 42(9), sept. 2009.

[13] A. Arsanjani, S. Ghosh, A. Allam, T. Abdollah, S. Gariapathy, and K. Holley. SOMA: a method for developing service-oriented solutions. *IBM Systems Journal*, 47:377–396, July 2008.

[14] G. Engels and M. Assmann. Service-oriented enterprise architectures: Evolution of concepts and methods. In *Enterprise Distributed Object Computing Conference, 2008. EDOC '08. 12th International IEEE*, pages XXXIV–XLIII. IEEE, September 2008.

[15] G. Engels, A. Hess, B. Humm, O. Juwig, M. Lohmann, J. Richter, M. Voß, and J. Willkomm. A method for engineering a true service-oriented architecture. In *International Conference on Enterprise Information System (ICEIS 2008)*, pages 272–281, Barcelona, Spain, 2008.

[16] T. Kohlborn, A. Korthaus, T. Chan, and M. Rosemann. Identification and analysis of business and software services-a consolidated approach. *IEEE Trans. Serv. Comput.*, 2(1):50–64, Jan 2009.

[17] V. Andrikopoulos, A. Bucchiarone, E. Di Nitto, R. Kazhamiakin, S. Lane, V. Mazza, and I. Richardson. Service engineering. In *S-CUBE Book*, pages 271–337, 2010.

[18] A. Kontogogos and P. Avgeriou. An overview of software engineering approaches to service oriented architectures in various fields. *IEEE International Workshops on Enabling Technologies*, 0:254–259, 2009.

[19] R. Boerner and M. Goeken. Service identification in soa governance literature review and implications for a new method. In *3rd IEEE International Conference on Digital Ecosystems and Technologies, 2009. DEST '09.*, pages 588–593, June 2009.

[20] M. F. Gholami, J. Habibi, F. Shams, and S. Khoshnevis. Criteria-based evaluation framework for service-oriented methodologies. In David Al-Dabass, Alessandra Orsoni, Richard Cant, and Ajith Abraham, editors, *UKSim*, pages 122–130. IEEE.

[21] E. Ramollari, D. Dranidis, and A. Simons. A survey of service oriented development methodologies. In *Proceedings of the 2nd European Young Researchers Workshop on Service Oriented Computing*, pages 75–80, June 2007.

[22] An American National Standard, editor. *A Guide to the project management body of knowledge.* Project Management Institute, Inc, 14 Campus Boulevard, Newtown Square, Pennsylvania 19073-3299 USA, 4 edition, 2008.

[23] L. Cernuzzi and G. Rossi. On the evaluation of agent oriented modeling methods. In *Proceedings of Agent Oriented Methodology Workshop*, pages 21–30, 2002.

[24] V. R. Basili and D. M. Weiss. A methodology for collecting valid software engineering data. *IEEE Transactions on Software Engineering*, SE-10(6):728–738, 1984.

[25] A. De Lucia and A. Qusef. Requirements engineering in agile software development. *Journal of Emerging Technologies in Web Intelligence*, 2(3):212–220, August 2010.

[26] K. Beck, A. Cockburn, R. Jeffries, and J. Highsmith. Agile manifesto, 2001.

[27] J. F. Abrantes and G. H. Travassos. Towards pertinent characteristics of agility and agile practices for software processes. *CLEI Electronic Journal*, 16:6 – 6, April 2013.

[28] K. Schwaber. *Agile Project Management With Scrum.* Microsoft Press, Redmond, WA, USA, 2004.

[29] P. Deemer and G. Benefield. The scrum primer. An introduction to agile project management with Scrum. 2007.

[30] S. Ambler. *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process*. John Wiley & Sons, Inc., New York, NY, USA, 2002.

[31] K. Beck and C. Andres. *Extreme Programming Explained: Embrace Change*. Addison-Wesley Professional, 2nd edition edition, 2004.

[32] P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta. Agile software development methods - review and analysis. Technical Report 478, VTT PUBLICATIONS, 2002.

[33] Chromatic. *Extreme programming pocket guide - team-based software development*. O'Reilly, 2003.

[34] J. Hurtado and C. Bastarrica. Implementing CMMI using a combination of agile methods. *CLEI Electron. J.*, 9(1), 2006.

[35] D. Turk, R. France, and B. Rumpe. Limitations of agile software processes. In *In proceedings of the Third International Conference on Extreme Programming and flexible processes in Software Engineering (XP2002)*, pages 43–46. Springer-Verlag, 2000.

[36] CMMI Product Team. CMMI for Development, Version 1.2. Technical report, Carnegie Mellon University, Software Engineering Institute, 2006.

[37] A. Erradi, S. Anand, and N. Kulkarni. SOAF: An architectural framework for service definition and realization. In *Proceedings of the IEEE International Conference on Services Computing*, SCC '06, pages 151–158, Washington, DC, USA, 2006. IEEE Computer Society.

[38] M. Papazoglou and W. Van Den Heuvel. Service oriented design and development methodology. *Int. J. Web Eng. Technol.*, 2:412–442, July 2006.

[39] D. X. Adamopoulos, G. Pavlou, and C.A. Papandreou. Advanced service creation using distributed object technology. *IEEE Communications Magazine*, 40(3):146–154, Mar 2002.

[40] M. Bell. *Service-Oriented Modeling. Service Analysis, Design and Architecture*. John Wiley and Sons, 2008.

[41] T. Böhmann and H. Krcmar. Modularisierung: Grundlagen und anwendung bei it-dienstleistungen. pages 45–84, 2005.

[42] S. Chang and S. Kim. A systematic approach to service-oriented analysis and design. In *PROFES*, pages 374–388, 2007.

[43] F. Chen, S. Li, H. Yang, C. Wang, and W.C.-C. Chu. Feature analysis for service-oriented reengineering. In *Software Engineering Conference, 2005. APSEC '05. 12th Asia-Pacific*, pages 201–208, Dec. 2005.

[44] C. Emig, J. Weisser, and S. Abeck. Development of soa-based software systems - an evolutionary programming approach. *Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICIW'06)*, 0:182, 2006.

[45] T. Erl. *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2005.

[46] A. Erradi, N. Kulkarni, and P. Maheshwari. Service design process for reusable services: Financial services case study. In *Proceedings of the 5th International Conference on Service-Oriented Computing*, ICSOC '07, pages 606–617, Berlin, Heidelberg, 2007. Springer-Verlag.

[47] B. Gold-Bernstein and W Ruh. *Enterprise Integration: The Essential Guide to Integration Solutions.* Addison Wesley Longman, 2004.

[48] R. Kaabi, C. Souveyet, and C. Rolland. Eliciting service composition in a goal driven manner. In *Proceedings of the 2Nd International Conference on Service Oriented Computing*, ICSOC '04, pages 308–315, New York, NY, USA, 2004. ACM.

[49] Y. Kim and K. Doh. The service modeling process based on use case refactoring. In Witold Abramowicz, editor, *Business Information Systems*, volume 4439 of *Lecture Notes in Computer Science*, pages 108–120. Springer Berlin Heidelberg, 2007.

[50] K. Klose, R. Knackstedt, and D. Beverungen. Identification of services - a stakeholder - based approach to soa development and its application in the area of production planning. In *Proceedings of the Fifteenth European Conference on Information Systems*, pages 1802–1814, 2007.

[51] F. Kohlmann and R. Alt. Business-driven service modelling - a methodological approach from the finance industry. In Witold Abramowicz and Leszek A. Maciaszek, editors, *BPSC*, volume 116 of *LNI*, pages 180–193. GI, 2007.

[52] C. Legner and R. Heutschi. Soa adoption in practice - findings from early soa implementations. In Hubert Österle, Joachim Schelp, and Robert Winter, editors, *ECIS*, pages 1643–1654. University of St. Gallen, 2007.

[53] E. A. Marks and M. Bell. *Service-Oriented Architecture. A Planning and Implementation Guide for Business and Technology.* John Wiley and Sons, 2006.

[54] B. Meyer and C. Mingins. Component-based development: from buzz to spark. *Computer*, 32(7):35–37, Jul 1999.

[55] E. Nadhan. Seven steps to a service-oriented evolution. *Business Integration Journal*, 1:41–43, 2004.

[56] D. Quartel, R. Dijkman, and M. van Sinderen. Methodological support for service-oriented design with ISDL. In *Proceedings of the 2nd International Conference on Service Oriented Computing*, ICSOC '04, pages 1–10, New York, NY, USA, 2004. ACM.

[57] J. Sewing, M. Rosemann, and M. Dumas. Process-oriented assessment of web services. *IJEBR*, 2(1):19–44, 2006.

[58] H.M. Sneed. Integrating legacy software into a service oriented architecture. In *Proceedings of the 10th European Conference on Software Maintenance and Reengineering, 2006. CSMR 2006*, pages 11–14, March 2006.

[59] Z. Stojanovic, A. Dahanayake, and H. Sol. Modeling and design of service-oriented architecture. In *2004 IEEE International Conference on Systems, Man and Cybernetics*, volume 5, pages 4147–4152, Oct 2004.

[60] V. Winkler. Identifikation und gestaltung von services. *Wirtschaftsinformatik*, 49(4):257–266, 2007.

[61] Z Zhang, R Liu, and H Yang. Service identification and packaging in service oriented reengineering. In *In Proceedings of the 7th International Conference on Software Engineering and Knowledge Engineering (SEKE)*, pages 241–249, 2005.

[62] O. Zimmermann, P. Krogdahl, and C. Gee. Elements of service-oriented analysis and design, June 2004.

[63] S. Lane and I. Richardson. Process models for service-based applications: A systematic literature review. *Inf. Softw. Technol.*, 53(5):424–439, May 2011.

[64] A. Anaby-Tavor, D. Amid, A. Sela, A. Fisher, K Zhang, and O. T. Jun. Towards a model driven service engineering process. In *IEEE Congress on Services - Part I, 2008*, pages 503–510, July 2008.

[65] A. Arsanjani and A. Allam. Service-oriented modeling and architecture for realization of an soa. In *Proceedings of the IEEE International Conference on Services Computing*, SCC '06, page 521, Washington, DC, USA, Sept. 2006. IEEE Computer Society.

[66] M. Autili, L. Berardinelli, V. Cortellessa, A. Di Marco, D. Di Ruscio, P. Inverardi, and M. Tivoli. A development process for self-adapting service oriented applications. In Bernd J. Krämer, Kwei-Jay Lin, and Priya Narasimhan, editors, *ICSOC*, volume 4749 of *Lecture Notes in Computer Science*, pages 442–448. Springer, 2007.

[67] L. Bahler, F. Caruso, and J. Micallef. A practical method and tool for systems engineering of service-oriented applications. In Boualem Benatallah, Fabio Casati, Dimitrios Georgakopoulos, Claudio Bartolini, Wasim Sadiq, and Claude Godart, editors, *Web Information Systems Engineering – WISE 2007*, volume 4831 of *Lecture Notes in Computer Science*, pages 472–483. Springer Berlin Heidelberg, 2007.

[68] A. Bercovici, F. Fournier, and A. J. Wecker. From business architecture to soa realization using mdd. In Ina Schieferdecker and Alan Hartman, editors, *ECMDA-FA*, volume 5095 of *Lecture Notes in Computer Science*, pages 381–392. Springer, 2008.

[69] V. Bicer, S. Lamparter, Y. Sure, and A. H. Dogru. Towards an interdisciplinary methodology for service-oriented system engineering. In *24th International Symposium on Computer and Information Sciences, 2009. ISCIS 2009*, pages 486–491, Sept. 2009.

[70] L. Bocchi, J. L. Fiadeiro, A. Lapadula, R. Pugliese, and R. Tiezzi. From architectural to behavioural specification of services. *Proceedings of the Sixth International Workshop on Formal Engineering approaches to Software Components and Architectures (FESCA 2009)*, 253(1):3–21, 2009.

[71] K. Boukadi, L. Vincent, and P. Burlat. Modeling adaptable business service for enterprise collaboration. In L. M. Camarinha-Matos, I. Paraskakis, and H. Afsarmanesh, editors, *Leveraging Knowledge for Innovation in Collaborative Networks*, volume 307 of *IFIP Advances in Information and Communication Technology*, pages 51–60. Springer Berlin Heidelberg, 2009.

[72] S. Chang and S. D. Kim. A service-oriented analysis and design approach to developing adaptable services. In *IEEE International Conference on Services Computing, 2007. SCC 2007*, pages 204–211, July 2007.

[73] W. Chengjun. Applying pattern oriented software engineering to web service development. In *International Seminar on Future Information Technology and Management Engineering, 2008. FITME '08.*, pages 214–217, Nov. 2008.

[74] I. Christou, S. Ponis, and E. Palaiologou. Using the agile unified process in banking. *IEEE Software*, 27, May 2010.

[75] E. Colombo, J. Mylopoulos, and P. Spoletini. Modeling and analyzing context-aware composition of services. In *Service-Oriented Computing - ICSOC 2005*, volume 3826 of *Lecture Notes in Computer Science*, pages 198–213. Springer Berlin Heidelberg, 2005.

[76] V. De Castro, E. Marcos, and R. Wieringa. Towards a service-oriented mda-based approach to the alignment of business processes with it systems: from the business model to a web service composition model. *Int. J. Cooperative Inf. Syst.*, 18(2):225–260, 2009.

[77] Y. Wautelet, Y. Achbany, J. Lange, and M. Kolp. A process for developing adaptable and open service systems: Application in supply chain management. In Joaquim Filipe and José Cordeiro, editors, *Enterprise Information Systems*, volume 24 of *Lecture Notes in Business Information Processing*, pages 564–576. Springer Berlin Heidelberg, 2009.

[78] B. Demchak, C. Farcas, E. Farcas, and I. H. Krüger. The treasure map for rich services. In *IRI*, pages 400–405. IEEE Systems, Man, and Cybernetics Society, 2007.

[79] M. Deubler, J. Grunbauer, G. Popp, G. Wimmel, and C. Salzmann. Tool supported development of service-based systems. In *11th Asia-Pacific Software Engineering Conference, 2004*, pages 99–108, Nov. 2004.

[80] M. Deubler, J. Grünbauer, G. Popp, G. Wimmel, and C. Salzmann. Towards a model-based and incremental development process for service-based systems. In *Proceedings of the IASTED International Conference on Software Engineering*, page 83–188, 2004.

[81] D. Dori. Soda: not just a drink! from an object-centered to a balanced object-process model-based enterprise systems development. In *Fourth and Third International Workshop on Model-Based Development of Computer-Based Systems and Model-Based Methodologies for Pervasive and Embedded Software, 2006. MBD/MOMPES 2006*, pages 12–14, March 2006.

[82] E. B. Fernandez, P. Cholmondeley, and O. Zimmermann. Extending a secure system development methodology to soa. In *18th International Workshop on Database and Expert Systems Applications, 2007. DEXA '07.*, pages 749–754, September 2007.

[83] H. Chen. Towards service engineering: Service orientation and business-it alignment. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences*, pages 114–114, Jan. 2008.

[84] Y. Howard and L. Kerschberg. A framework for dynamic semantic web services management. *International Journal of Cooperative Information Systems, Special Issue on Service Oriented Modeling*, 13:441–485, 2004.

[85] R. Klischewski and R. Abubakr. Can e-Government adopters benefit from a technology-first approach?. The case of Egypt embarking on service-oriented architecture. In *2010 43rd Hawaii International Conference on System Sciences (HICSS)*, pages 1–10, Jan. 2010.

[86] A. Ivanyukovich, G. R. Gangadharan, V. D'Andrea, and M. Marchese. Towards a service-oriented development methodology. *J. Integr. Des. Process Sci.*, 9(3):53–62, July 2005.

[87] S. K. Johnson and A. W. Brown. A model-driven development approach to creating service-oriented solutions. In *Proceedings of the 4th International Conference on Service-Oriented Computing*, ICSOC'06, pages 624–636, Berlin, Heidelberg, 2006. Springer-Verlag.

[88] S. Kambhampaty. Service oriented analysis and design process for the enterprise. In *Proceedings of the 7th Conference on 7th WSEAS International Conference on Applied Computer Science*, volume 7 of *ACS'07*, pages 366–371, Stevens Point, Wisconsin, USA, 2007. World Scientific and Engineering Academy and Society (WSEAS).

[89] M. Karam, S. Dascalu, H. Safa, R. Santina, and Z. Koteich. A product-line architecture for web service-based visual composition of web applications. *J. Syst. Softw.*, 81(6):855–867, Jun. 2008.

[90] H. Karhunen, M. Jantti, and A. Eerola. Service-oriented software engineering (sose) framework. In *Proceedings of ICSSSM '05. 2005 International Conference on Services Systems and Services Management, 2005*, volume 2, pages 1199–1204, June 2005.

[91] T. Karle and A. Oberweis. Collaborative model driven software development for soa-based systems. In Roland Kaschek, Christian Kop, Claudia Steinberger, and Günther Fliedl, editors, *Information Systems and e-Business Technologies*, volume 5 of *Lecture Notes in Business Information Processing*, pages 189–200. Springer Berlin Heidelberg, 2008.

[92] A. Kenzi, B. El Asri, M. Nassar, and A. Kriouile. A model driven framework for multiview service oriented system development. In *Computer Systems and Applications, 2009. AICCSA 2009. IEEE/ACS International Conference on*, pages 404–411, May. 2009.

[93] J. Koehler, R. Hauser, J. Küster, K. Ryndina, J. Vanhatalo, and M. Wahler. The role of visual modeling and model transformations in business-driven development. *Electron. Notes Theor. Comput. Sci.*, 211:5–15, Apr. 2008.

[94] I. H. Kruger and R. Mathew. Systematic development and exploration of service-oriented software architectures. In *Proceedings of Fourth Working IEEE/IFIP Conference on Software Architecture, 2004. WICSA 2004*, pages 177–187, June 2004.

[95] I. Kürger, M. Meisinger, and M. Menarini. Applying service-oriented development to complex systems: Bart case study. In Fabrice Kordon and Janos Sztipanovits, editors, *Reliable Systems on Unreliable Networked Platforms*, volume 4322 of *Lecture Notes in Computer Science*, pages 26–46. Springer Berlin Heidelberg, 2007.

[96] S. Lamparter and Y. Sure. An interdisciplinary methodology for building service-oriented systems on the web. *2013 IEEE International Conference on Services Computing*, 2:475–478, 2008.

[97] M. López-Sanz, C. J. Acuòa, E. Cuesta, and E. Marcos. Modelling of service-oriented architectures with uml. *Electron. Notes Theor. Comput. Sci.*, 194(4):23–37, April 2008.

[98] M. Meisinger and I. H. Kruger. A service-oriented extension of the v-modell xt. In *14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems, 2007. ECBS '07.*, pages 256–268, March 2007.

[99] S. Mittal. *Devs Unified Process for Integrated Development and Testing of Service Oriented Architectures.* PhD thesis, Tucson, AZ, USA, 2007.

[100] B. Orriens, Jian Y., and M. Papazoglou. A rule driven approach for developing adaptive service oriented business collaboration. In *IEEE International Conference on Services Computing, 2006. SCC '06.*, pages 182–189, Sept. 2006.

[101] C. Pahl. Semantic model-driven architecting of service-based software systems. *Inf. Softw. Technol.*, 49(8):838–850, August 2007.

[102] N. Protogeros, D. Tektonidis, A. Mavridis, C. Wills, and A. Koumpis. Fuse: A framework to support services unified process. In *Enterprise Interoperability III: New Challenges and Industrial Approaches*, pages 209–220. Springer, 2008.

[103] Z. Ren, B. Jin, and J. Li. A new web application development methodology: Web service composition. In *International Workshop on Web Services, E-Business, and the Semantic Web (WES 2003)*, volume 3095 of *Lecture Notes in Computer Science*, pages 134–145. Springer Berlin Heidelberg, 2004.

[104] M. Rychlý and P. Weiss. Modeling of service oriented architecture - from business process to service realisation. In *International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2008)*, pages 140–146. INSTICC Press, 2008.

[105] C. Soo Ho. A systematic analysis and design approach to develop adaptable services in service oriented computing. In *IEEE SCW*, pages 375–378, 2007.

[106] J. K. Strosnider, P. Nandi, S. Kumaran, S. Ghosh, and A. Arsnajani. Model-driven synthesis of soa solutions. *IBM Systems Journal*, 47(3):415–432, July 2008.

[107] S. Vale and S. Hammoudi. Model driven development of context-aware service oriented architecture. In *11th IEEE International Conference on Computational Science and Engineering Workshops, 2008. CSEWORKSHOPS '08*, pages 412–418, July 2008.

[108] H. Wada, J. Suzuki, and K. Oba. A feature modeling support for non-functional constraints in service oriented architecture. In *IEEE International Conference on Services Computing, 2007. SCC 2007.*, pages 187–195, July 2007.

[109] M. Wirsing, M. Hölzl, L. Acciai, F. Banti, A. Clark, A. Fantechi, S. Gilmore, S. Gnesi, L. Gönczy, N. Koch, A. Lapadula, P. Mayer, F. Mazzanti, R. Pugliese, A. Schroeder, F. Tiezzi, M. Tribastone, and D. Varró. Sensoria patterns: Augmenting service engineering with formal analysis, transformation and dynamicity. In *Leveraging Applications of Formal Methods, Verification and Validation*, volume 17 of *Third International Symposium*, pages 170–190. Springer Berlin Heidelberg, 2008.

[110] M Wirsing, M. Hölzl, N. Koch, P. Mayer, and A. Schroeder. Service engineering: The sensoria model driven approach. In *Proceedings of Software Engineering Research, Management and Applications (SERA 2008)*. IEEE Computer Society.

[111] P. Baxter and S. Jack. Qualitative case study methodology: Study design and implementation for novice researchers. *The Qualitative Report*, 13(4):544–559, dec 2008.

[112] K. M. Eisenhardt. Building theories from case study research. *The Academy of Management Review*, 14(4):532–550, Oct. 1989.

[113] Y. Tian, Y. Su, and X. Zhuang. Research on service identification methods based on SOA. In *2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)*, volume 6, pages V6–27 – V6–31, Aug. 2010.

[114] N. Zhou, L. Zhang, Y. Chee, and L. Chen. Legacy asset analysis and integration in model-driven soa solution. In *2010 IEEE International Conference on Services Computing (SCC)*, SCC '10, pages 554–561, Washington, DC, USA, july 2010.

[115] L. Zhang, A. Arsanjani, A. Allam, D. Lu, and Y. Chee. Variation oriented analysis for SOA solution design. In *IEEE SCC*, pages 560–568. IEEE Computer Society, 2007.

[116] K. Levi and A. Arsanjani. A goal-driven approach to enterprise component identification and specification. *Commun. ACM*, 45(10):45–52, Oct 2002.

[117] W. T. Tsai, X. Zhou, Y. Chen, B. Xiao, R. Paul, and W. Chu. Roadmap to a full service broker in service-oriented architecture. In *Proceedings of the IEEE International Conference on e-Business Engineering*, ICEBE '07, pages 657–660, Washington, DC, USA, 2007. IEEE Computer Society.

[118] J. Gurevitch and L. Hedges. *Meta-analysis: Combining results of independent experiments*, chapter Design and Analysis of Ecological Experiments, pages 347—-369. Oxford University Press, Oxford, 2001.

[119] L. V. Hedges and I. Olkin. *Statistical methods for meta-analysis.* Academic press, San Diego, 1985.

[120] J. Miller. Applying meta-analytical procedures to software engineering experiments. *Journal of Systems and Software*, 54:29–39, 1998.

[121] G. V. Glass. Primary, secondary, and meta-analysis of research. *Educational Researcher*, 5(10):3–8, 1976.

[122] IT Governance Institute. *Cobit 4.1.* ISA, 2007.

[123] P. Baxter and S. Jack. Qualitative case study methodology: Study design and implementation for novice researchers. *The Qualitative Report*, 13(4):544–559, dec 2008.

[124] K. M. Eisenhardt. Building theories from case study research. *The Academy of Management Review*, 14(4):532–550, Oct. 1989.

Appendix A

Questions associated to the Tree of Goals

In this appendix, we propose the content of the whole questionnaire that is composed of the questions associated to each goal leaf of the tree of goals (TOG) (see Chapter 4). The whole TOG is shown in Table A.1, and it fixes the scope of the assessment. Thus, we can prune goals or dimensions of the TOG to reduce the scope.

Table A.1: Tree of Goals

SO	**Service Oriented**
SO.1	Service Description
SO.2	Service Implementation
SO.3	Exchange Standards
SE	**Software Engineering**
SE.1	Life Cycle
SE.1.1	Requirement
SE.1.2	Analysis and Design
SE.1.3	Development and Deployment
SE.2	Quality Assurance
SE.2.1	Quality Management
SE.2.2	Configuration and Change Management
SE.2.3	Test
PM	**Project Management**
PM.1	Development Management
PM.2	Communication Management
PM.3	Human Resource Management
PM.3.1	Recruitment of Human Resources
PM.3.2	Role Assignment
PM.3.3	Monitoring and Controlling Human Resource
PM.4	Procurement management
SU	**Sustainability**
SU.1	Documentation
SU.2	Implementation Cost
SU.3	Easiness

Questionnaire

The questions associated to each goal-leaf are contained in one questionnaire. The TOG's pruning can vary the number of questions of the questionnaire, as a result, the scope of assessment of one case study.

A copy of the questionnaire is filled in by each interviewee related with a case study.

The whole questionnaire is described in the next sub-section.

SO Service Oriented

SO.1 Service Description

1. Are the services relationships shown?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
2. Are the services roles shown?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
3. Are the services inputs shown?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
4. Are the services outputs shown?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
5. Is the Service Level Agreement (SLA) carried out?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely

- ◯ Normally
- ◯ Always

6. Is the services composition described?

   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

SO.2 Service Implementation

1. Is there a service repository?

   - ◯ Does not Apply
   - ◯ NO
   - ◯ YES

2. Was enterprise service bus implemented?

   - ◯ Does not Apply
   - ◯ NO
   - ◯ YES

3. Is the service composition carried out using Business Process Execution Language?

   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

SO.3 Exchange Standards

1. Was a standardized process for data exchange defined?

   - ◯ Does not Apply
   - ◯ NO
   - ◯ YES

2. Was a data format for data exchange defined?

   - ◯ Does not Apply
   - ◯ NO
   - ◯ YES

3. Was a method for concepts unification defined?

   - ◯ Does not Apply
   - ◯ NO
   - ◯ YES

SE Software Engineering

SE.1 Life Cycle

SE.1.1 Requirement

1. Were the requirements described?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
2. Were the requirements gathered from the users?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
3. Was the priority list created?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
4. Was the development priority fixed with users?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
5. Were the use cases created?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
6. Were the mock-up created?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

SE.1.2 Analysis and Design

1. Were the Business Models created?

   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

2. Was the domain model created?

   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

3. Was the structure model created?

   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

4. Was the data model created?

   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

5. Were the service models created?

   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

SE.1.3 Development and Deployment

1. Was the deployment model created?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

2. Was the service composition implemented?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

3. Was the deployment plan created?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

4. Was the back up plan created?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

5. Was the coding standard met?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

6. Was the database standard met?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally

- ◯ Always

7. Was the interface standard met?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

8. Were the services published in the service repository?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

SE.2 Quality Assurance

SE.2.1 Quality Management

1. Were the development processes defined?
   - ◯ Does not Apply
   - ◯ NO
   - ◯ YES

2. Were the maintenance processes defined?
   - ◯ Does not Apply
   - ◯ NO
   - ◯ YES

3. Was the deployment plan created?
   - ◯ Does not Apply
   - ◯ NO
   - ◯ YES

4. Were the templates for each work-product defined?
   - ◯ Does not Apply
   - ◯ NO
   - ◯ YES

5. Was the coding standard defined?
   - ◯ Does not Apply
   - ◯ NO
   - ◯ YES

6. Was the database standard defined?

   ◯ Does not Apply
   ◯ NO
   ◯ YES

7. Was the interface standard defined?

   ◯ Does not Apply
   ◯ NO
   ◯ YES

8. Was the user manual created?

   ◯ Does not Apply
   ◯ NO
   ◯ YES

SE.2.2 Configuration and Change Management

1. Is there a responsible for configuration?

   ◯ Does not Apply
   ◯ NO
   ◯ YES

2. Was the configuration plan created?

   ◯ Does not Apply
   ◯ NO
   ◯ YES

3. Was a revision control system (RCS) installed?

   ◯ Does not Apply
   ◯ NO
   ◯ YES

4. Have the project assets included in RCS?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

5. Was the change request process defined?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely

- ◯ Normally
- ◯ Always

6. Was the change request document defined?

- ◯ Does not Apply
- ◯ NO
- ◯ YES

SE.2.3 Test

1. Were there tester (it can be one person or more)?

- ◯ Does not Apply
- ◯ Never
- ◯ Rarely
- ◯ Normally
- ◯ Always

2. Was the software verified by tester?

- ◯ Does not Apply
- ◯ Never
- ◯ Rarely
- ◯ Normally
- ◯ Always

3. Was the software validated by tester?

- ◯ Does not Apply
- ◯ Never
- ◯ Rarely
- ◯ Normally
- ◯ Always

4. Is there a environment for testing?

- ◯ Does not Apply
- ◯ Never
- ◯ Rarely
- ◯ Normally
- ◯ Always

5. Is the test environment different from the production environment?

- ◯ Does not Apply
- ◯ Never
- ◯ Rarely
- ◯ Normally

- ◯ Always

6. Was the software verified by users?
    - ◯ Does not Apply
    - ◯ Never
    - ◯ Rarely
    - ◯ Normally
    - ◯ Always

7. Was the software validated by users?
    - ◯ Does not Apply
    - ◯ Never
    - ◯ Rarely
    - ◯ Normally
    - ◯ Always

8. Did the users sign the document of software acceptance?
    - ◯ Does not Apply
    - ◯ Never
    - ◯ Rarely
    - ◯ Normally
    - ◯ Always

PM Project Management

PM.1 Development Management

1. Was the project scope defined?
    - ◯ Does not Apply
    - ◯ NO
    - ◯ YES

2. Was the project scope controlled?
    - ◯ Does not Apply
    - ◯ NO
    - ◯ YES

3. Was the project scope updated when it changes?
    - ◯ Does not Apply
    - ◯ Never
    - ◯ Rarely
    - ◯ Normally
    - ◯ Always

4. Was the project duration estimated?

   ◯ Does not Apply
   ◯ NO
   ◯ YES

5. Was the project duration controlled?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

6. Was the project duration updated when it changes?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

7. Was the project cost estimated?

   ◯ Does not Apply
   ◯ NO
   ◯ YES

8. Was the project cost controlled?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

9. Was the project cost updated when it changes?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

10. Was the project plan created?

    ◯ Does not Apply
    ◯ NO
    ◯ YES

11. Was the project plan updated when it changes?

    ◯ Does not Apply
    ◯ Never
    ◯ Rarely
    ◯ Normally
    ◯ Always

12. Was the schedule created?

    ◯ Does not Apply
    ◯ NO
    ◯ YES

13. Was the schedule controlled?

    ◯ Does not Apply
    ◯ Never
    ◯ Rarely
    ◯ Normally
    ◯ Always

14. Was the schedule updated when it changes?

    ◯ Does not Apply
    ◯ Never
    ◯ Rarely
    ◯ Normally
    ◯ Always

15. Were the iterations planned?

    ◯ Does not Apply
    ◯ NO
    ◯ YES

PM.2 Communication Management

1. At the beginning of the project, was the meeting frequency defined?

    ◯ Does not Apply
    ◯ NO
    ◯ YES

2. Were the meetings carried out according to the frequency established?

    ◯ Does not Apply
    ◯ Never
    ◯ Rarely

- ◯ Normally
- ◯ Always

3. Were the meeting minutes elaborated?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

4. Was the user training envisaged?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

PM.3 Human Resource Management

PM.3.1 Recruitment of Human Resources

1. Was a recruitment process defined?
   - ◯ Does not Apply
   - ◯ NO
   - ◯ YES

2. Was the new staff notified about its tasks?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

3. Was the new staff notified about the organization procedure related with its tasks?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

PM.3.2 Role Assignation

1. Were the roles defined?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

2. Did the staff know their responsibilities?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

PM.3.3 Monitoring and Controlling Human Resource

1. Were the tasks assigned to the staff?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

2. Was there a responsible for controlling the staff?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

3. Were the tasks execution controlled?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

PM.4 Procurement Management

1. Was there a standardized process to call providers?

   ◯ Does not Apply
   ◯ NO
   ◯ YES

2. Were the providers evaluated?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

3. Was a contract with provider created?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

4. Was a process for tracking providers created?

   ◯ Does not Apply
   ◯ NO
   ◯ YES

5. Was the product delivered by the provider verified?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

6. Was the product delivered by the provider validated?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

SU Sustainability

SU.1 Documentation

1. Were there documents to describe the requirements?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

2. Were the requirements description updated in case of changes?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

3. Were there documents to describe the architecture?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

4. Were the architecture description updated in case of changes?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

SU.2 Implementation Cost

1. Were strategies to reduce costs defined?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

2. Are there open source tools to elaborate the models and documents?

   ◯ Does not Apply
   ◯ Never
   ◯ Rarely
   ◯ Normally
   ◯ Always

SU.3 Easiness

1. Were the staff trained to use the method?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
2. Are there open source tools to elaborate the models and documents?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
3. Were the staff trained to use the method?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
4. Were the creation of the models easy for the staff?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always
5. Were the use of the templates easy for the staff?
   - ◯ Does not Apply
   - ◯ Never
   - ◯ Rarely
   - ◯ Normally
   - ◯ Always

Appendix B

Surveys in the Public Sector of Paraguayan Government

The surveys aim to obtain information about ICT situation in public agencies of Paraguayan Government. The agencies involved are ministries, national insurance, public companies, and secretaries.

They encompass the topics of: ICT teams composition; ICT communication management; methodologies of development and maintenance; and, ICT staff skills.

B.1 Survey about Software Development

It encompasses the topics of: ICT teams composition; ICT communication management; and, methodologies of development and maintenance. It was sent to the participant through a tool: http://www.esurveyspro.com/.

The representatives of 11 institutions have answered the survey. Among these institutions there are:

- 6 Ministries
- 1 Welfare Payment
- 1 Ministry Dependency
- 1 Central Government Dependency
- 2 Government Enterprises

The next sections describe the survey content.

B.1.1 ICT Team Composition

The section tackles size, composition, and location of teams in public agencies. It describes questionnaires, answers, and summary.

How many members is normally team composed of?

Generally, the teams are small, not more than 5 members, (40%), but also there are high percentage (30%) of teams with more than 50 members.

Is the staff geographically distributed?

The staff is normally located in different places (70%) while only 30% are in the same building.

Are the roles defined?

The roles are defined, and a person can play various roles at the same time in the same project.

How development is it carried out?

Among interviewees the 50% of developments are performed by the team of agency, the 41% are outsourced, and 6% are packages. It is important to note that the development can be performed blending outsource with in-house (50%). Moreover, only the 30% of development are carried out in house, and 10% only use outsource while the mix among outsource, in-house and packages is of 10%.

B.1.2 ICT Communication Management

How often is the team meetings?

The frequency of team meetings are normally many times a month (60%) or once (40%), but not daily.

How often is the meetings with users?

The meetings with users are as follows:

- At the beginning of development, 21%.
- During development, 28%.
- For testing, 31%.
- To delivery, 17%.
- Never, 3%.

The meetings are arranged following the next patterns (see Figure B.1):

- At the beginning of development, during development, testing, delivery, 56%.
- At the beginning of development, during development, testing, 22%.
- During development, testing, delivery, 11%.
- Testing, 11%.

Do you think the meeting frequency is appropriate, insufficient, or redundant?

The majority of teams think the meetings frequency are appropriate 73% while the 27% believes it is insufficient.

How do developers receive the specification of functionality to develop?

The developers can obtain the specification of the functionalities through documents and orally (63%). Moreover, they receive in the 18% of cases only either orally or documents.

B.1.3 Methodology of Development and Maintenance

Which methodologies do you utilize in the institution?

The majority of teams use waterfall (60%), but only the (60%) are satisfied with it. The agile methods are applied by 20% including Agile Unified Process (AUP) and Extreme Programming (XP), and they satisfy the team. None method is followed by 20%.

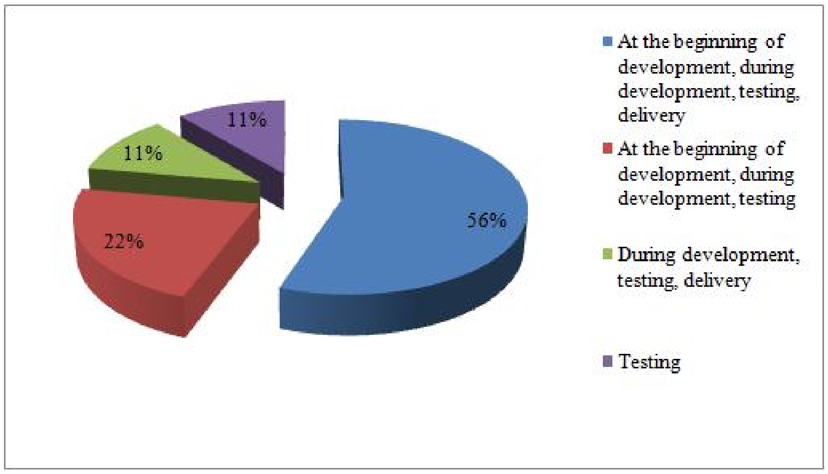


Figure B.1: Meetings Pattern

Which activities does the development process tackle?

The development process normally tackles getting requirements (25%), analysis, and design (41%), project management (17%), and outsourcing management (17%).

Which is the percentage of development time that the documentation elaboration has taken?

Less than the 10% of development time is used to make documentation by 40% of teams, and the same percentage of them utilize between 10% and 30%. Only the 20% of teams employ more than the 30%, and less than the 50% for documenting.

Which is the percentage of development time that testing has taken?

Testing takes between 10% and 30% development time in the 60% of teams while it consumes more than the 30%, and less than the 50% in the 20% of them. Less than the 10% of time is used for testing in 20% of teams.

Which is the percentage of legacy systems that have documentation?

Less than 50% of legacy systems have documentation, and this number is distributed as follows:

- In the 60% of agencies between 30% and 50% of legacy systems have documentation.
- In the 20% of agencies between 10% and 30% of legacy systems have documentation.
- In the 20% of agencies less than 10% of legacy systems have documentation.

Are systems documentation updated when design changes?

In the 20% of the cases, the documentation is either never, or normally updated while it is rarely brought up to date in the 60%.

Which are the modelling languages used in the institution?

The 55% of interviewees mentioned that they have not used modelling languages while the 27% employ UML (Unified Modelling Language), and BPMN (Business Process Modelling Notation) 18%. Among teams that utilize modelling language the 25% of them blend UML, and BPMN. However, the non hybrid implementations are UML 50%, BPMN 25%.

Which are the UML diagrams used in the institution?

The percentage in which each UML diagram is used it is as follows:

- 6% Sequence Diagram.
- 6% Component Diagram.
- 6% Object Diagram.
- 6% State Machine.
- 6% Interaction Diagram.
- 17% Use Case.
- 12% Activity Diagram.
- 17% Class Diagram.
- 12% Business Case Diagram.

Various diagrams can applied together. Thus, the following combinations have been reported.

- Use Case Diagram, Business Case Diagram, Sequence Diagram, Class Diagram, Component Diagram, and Activity Diagram.
- Use Case Diagram, Business Case Diagram, Sequence Diagram, Class Diagram, Object Diagram, and Interaction Diagram.
- Use Case Diagram, Class Diagram, and Activity Diagram.

Note that in all the cases the use case, and class diagrams are employed.

Are there other types of diagrams used in the institution apart from the UML diagrams?

The 36% of teams reported that the they utilize Entity Relation Diagram, and the 21% Flow Chart Diagram. Both are utilized together by the 27% of teams and only Entity Relation Diagram is implemented by the 18%.

The percentage of teams that do not use neither of them is of 43%.

Which are metrics used to measure either size or complexity of developments?

Non metric is employed by teams.

Which are quality standards used?

Non quality standard is utilized by teams.

Has the Test-driven development (TDD) helped to improve software quality?

TDD is utilized only by the 18% of teams.

Do you use techniques or criteria to filter functionalities that will be implemented as Web services?

The Web services are not widely used. They are used only in the 40% of cases.

B.2 ICT Staff Skill

The data presented in Section B.2 given by developers of diverse dependencies of 3 ministries, and the welfare payment. It focuses on finding out the know-how of developers about modelling languages. The data was collected during a training course where the developers filled in their responses in paper. The responses are presented in the next sections.

B.2.1 Which modelling language do you know?

The 91% of developers know UML, and the 18% of them BPMN while 9% neither of them.

B.2.2 Which diagrams do you know?

The 91% of developers know class diagram, and the 82% of them Entity Relation Diagram while 9% neither of them.

B.2.3 Do you know XSD (XML Schema Definition Language) or DTD (Document type definition)?

The 55% of developers know XSD while DTD by neither of them.

B.2.4 Do developers receive documents describing functionalities to program?

The developers have admitted that:

- They may receive functionalities specification 64%.
- They always receive specification 18%.
- They never receive specification 18%.

B.2.5 Do you think that is easy to understand the functionality to develop if you have its documentation?

The 91% of developers mention that they would like to have specification before programming.