D 2016

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

EQUALPI: A FRAMEWORK TO EVALUTE THE QUALITY OF THE IMPLEMENTATION OF THE CMMI PRACTICES

ISABEL DE JESUS LOPES MARGARIDO

TESE DE DOUTORAMENTO APRESENTADA
À FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO EM
ENGENHARIA INFORMÁTICA

Faculdade de Engenharia da Universidade do Porto

EQualPI: a Framework to Evaluate the Quality of the Implementation of the CMMI Practices

Isabel de Jesus Lopes Margarido

Programa Doutoral em Engenharia Informática

Advisor: Raul Moreira Vidal, FEUP

Co-advisor: Marco Paulo Amorim Vieira, FCTUC

Porto, November 7, 2016

EQualPI: a Framework to Evaluate the Quality of the Implementation of the CMMI Practices

Isabel de Jesus Lopes Margarido

Programa Doutoral em Engenharia Informática

Approved by unanimity:

President: Doutor Eugénio da Costa Oliveira, Professor Catedrático da FEUP

Referee: Doutor David Zubrow, Associate Director of Empirical Research, Software Solutions Division, Software Engineering Institute, Carnegie Mellon University

Referee: Doutor Marco Paulo Amorim Vieira, Professor Associado com Agregação do Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra (Coorientador e especialista em área científica distinta)

Referee: Doutor Fernando Manuel Pereira da Costa Brito e Abreu, Professor Associado do Departamento de Ciências e Tecnologias de Informação do ISCTE-IUL

Referee: Doutor Raul Fernando de Almeida Moreira Vidal, Professor Associado do Departamento de Engenharia Informática da Faculdade de Engenharia da Universidade do Porto (Orientador)

Referee: Doutor João Carlos Pascoal Faria, Professor Auxiliar do Departamento de Engenharia Informática da Faculdade de Engenharia da Universidade do Porto

Referee: Doutora Ana Cristina Ramada Paiva, Professora Auxiliar do Departamento de Engenharia Informática da Faculdade de Engenharia da Universidade do Porto

Porto, December 7, 2016

Abstract

The Capability Maturity Model Integration® (CMMI) allows organisations to improve the quality of their products and customer satisfaction; reduce cost, schedule, and rework; and make their processes more predictable. However, this is not always the case, as there are differences in performance between CMMI organisations, depending not only on the context of the business, projects, and team, but also on the methodologies used in the implementation of the model practices. CMMI version 1.3 is more focused on the performance of the organisations than previous versions. However, the Standard CMMI Appraisal Method for Process Improvement^{SM} (SCAMPI) is not focused on evaluating performance.

To evaluate practices performance it is necessary to consider the goal of executing the practice, and the quality of implementation of a practice is reflected in its outputs. Therefore, if we can establish a relationship between the methods used to implement a practice and the performance of its results, we can use such relationship in a framework to evaluate the quality of implementation of the practice. We consider that it is possible to objectively measure the quality of implementation of CMMI practices by applying statistical methods in the analysis of organisations' data, in order to evaluate process improvement initiatives and predict their impact on organisational performance.

In this research we develop a framework to evaluate the quality of the implementation of the CMMI practices that supports the comparison of the quality of the implementation before and after improvements are put in place. Considering the extent of the CMMI model, we demonstrate the framework in the Project Planning's Specific Practice 1.4 "Estimate Effort and Cost". We consider that the quality of implementation of this practice is measured by the Effort Estimation Accuracy, defined by a set of controllable and uncontrollable factors, and it can be improved by acting on the controllable factors. To implement and validate our framework we conducted literature reviews, case studies on high maturity organisations, data analysis of a survey performed by the Software Engineering Institute (SEI) and on the Team Software Process^{SM} (TSP) Database, which we used to build a regression model, and conducted an experiment with students to define a process improvement.

This Ph.D. thesis provides to software development organisations a framework for self-assessing the quality of the implementation of the CMMI practices, EQualPI. The framework is also useful to the CMMI Institute, in order to evaluate the performance of the organisations from one SCAMPI A to the next. The framework is already populated with recommendations to support organisations willing to implement CMMI to avoid a set of problems and difficulties, factors to consider when implementing Measurement and Analysis for CMMI high maturity levels, a procedure based on the scientific method to conduct process improvements, a performance indicator model to evaluate the quality of implementation of the effort estimation process, and indicators related with effort estimation accuracy. Additionally, with the implementation and validation of the EQualPI framework, we provide the procedure we used to analyse data from the SEI TSP Database and define process variables, and by applying the process improvements procedure, we contribute with a defects classification specific for requirements.

Resumo

O Capability Maturity Model Integration® (CMMI) permite às organizações melhorar a qualidade dos seus produtos e satisfação dos seus clientes; reduzir custos, calendário e a necessidade de refazer trabalho. Com o CMMI os processos passam a ser mais previsíveis. No entanto nem sempre é este o caso, dado que há uma diferença de desempenho entre organizações que usam CMMI, que depende não somente do negócio, projectos e equipas mas também das metodologias usadas na implementação das práticas modelo. A versão 1.3 do CMMI é mais focada na performance das organizações do que as anteriores, no entanto o seu método de avaliação, Standard Appraisal Method for Process Improvement ^{SM} (SCAMPI), não tem como objectivo avaliar o desempenho das organizações.

Para avaliar o desempenho de práticas é necessário considerar o objectivo de as executar e que a qualidade de implementação de uma prática se reflecte nos seus resultados. Por esse motivo, podemos estabelecer uma relação entre os métodos utilizados na implementação de uma prática e os resultados da sua performance. Consideramos que é possível medir objectivamente a qualidade de implementação das práticas do CMMI aplicando métodos estatísticos na análise dos dados de organizações, para dessa forma avaliar as iniciativas de melhoria de processos e prever o impacto que essas melhorias vão ter na performance da organização.

Nesta investigação científica desenvolvemos uma *framework* para avaliar a qualidade de implementação das práticas CMMI que permite comparar a qualidade da implementação antes e depois de introduzir uma melhoria. No entanto, o CMMI é extenso, por esse motivo vamos demonstrar a *framework* na área específica do processo de Planeamento de Projectos 1.4 "Estimar esforço e custo". Consideramos que a qualidade de implementação desta prática é medida através da precisão da estimativa de esforço, definida por um conjunto de factores controláveis e não controláveis, e que o seu valor pode ser melhorado actuando sobre os factores controláveis. Para implementar e validar a nossa framework efectuámos revisões de literatura, casos de estudo em organizações de alta maturidade, análises de dados sobre um inquérito realizado pelo *Software Engineering Institute* (SEI) e sobre a base de dados do Team Software Process^{SM} (TSP), que utilizámos para implementar um modelo de regressão linear, e conduzimos uma experiência com estudantes para definir uma melhoria de processo.

Como resultado desta tese de Doutoramento disponibilizamos às organizações a EQualPI, uma *framework* de auto-avaliação da qualidade de implementação das práticas CMMI. Esta *framework* também é útil para o CMMI Institute poder avaliar o desempenho das organizações aquando da recertificação. A EQualPI tem já incluídas recomendações de suporte às organizações que pretendem implementar o CMMI evitando um conjunto de problemas e dificuldades, recomendações sobre factores a considerar quando se implementa a prática de *Measurement and Analysis* em níveis de alta maturidade, um procedimento de melhoria de processo baseado no método científico, um modelo de um indicador de performance para avaliar a qualidade de implementação da prática de estimação de esforço, bem como um conjunto de indicadores relacionados com a precisão da estimação de esforço. Adicionalmente, da implementação e validação da EQualPI,

resultou o procedimento que seguimos na análise dos dados TSP que se encontram na base de dados do SEI na definição das variáveis de processo, da aplicação do procedimento de melhorias de processo resultou também uma lista de classificação de defeitos específica para documentos de requisitos.

Acknowledgements

This adventure would have not been possible without the love and support of my better half, who took care of me in all the hard moments; my parents and my sisters who are always there for me and understood my long absences to do this research; my native reviewer and Mena.

I thank the SEI for receiving me so well, in particular Paul Nielsen, Anita Carlton, Rusty Young, Eileen Forrester, Gene Miluk, Jim Over, Mike Conrad, Bob Stoddard and Jim McCurley. Special thanks to Dave Zubrow and Bill Nichols for the great work we did together, and Dennis Goldenson for his advice and support.

I thank my SEPG friends, including Mike Campo, Kees Hermus and Mia, for the great moments we spent together and CMMI talks.

My PhD colleagues, Professors and CISUC colleagues for all discussions and good times.

A big thank you to my MBFs, for all the hangouts and my friends, close and absent, for supporting me.

I also thank my supervisors, co-authors and reviewers, who contributed to this research.

Finally, I cannot finish without thanking to the person without whom I could not have done this research, Watts Humphrey.

Isabel de Jesus Lopes Margarido

This research was partially funded by Fundação para a Ciência e a Tecnologia (FCT) and Critical Software S.A., support grant SFRH/BDE/33803/2009:

Contents

List of Figures

List of Tables

Acronyms and Definitions

Acronyms

AIM	Accelerated Improvement Method
AFP	Automated Function Points
BSC	Balanced Score Card
BUn	Basic Unit
BU	Business Unit
CAR	Causal Analysis and Resolution (process area)
CI	Configuration Item
CI, II	COCOMO I, COCOMO II
CISQ	Consortium of IT Software Quality
CL	Capability Level
CM	Configuration Management (process area)
CMM	Capability Maturity Model
CMMI	Capability Maturity Model Integration
CMMI-ACQ	CMMI for Acquisition
CMMI-DEV	CMMI for Development
CMMI-SVC	CMMI for Services
COCOMO	Constructive Cost Model
CODEINSP	Code Inspection
CR	Code Review
DAR	Decision Analysis and Resolution (process area)
DCA	Defect Causal Analysis
DL	Deliverable
DLD	Detailed Design
DLDR	Detailed Design Review
DMAIC	Define Measure Analyse Implement Control
DoD	Department of Defence (funding the SEI)
DOE	Design of Experiments
DPPI	Defect Prevention Based Process Improvement
EEA	Effort Estimation Accuracy
EQualPI	Framework to Evaluate the Quality of Process Improvements
f	Function
FCT/UNL	Faculty of Science and Technology, Universidade NOVA de Lisboa
FCTUC	Faculty of Sciences and Technology, University of Coimbra
FEUP	Faculty of Engineering, University of Porto
FI	Fully Implemented
FL	Fuzzy Logic
FPA	Function Point Analysis

FSS	Feature Subset Selection
GA	Genetic Algorithm
GG	Generic Goal
GP	Genetic Programming
GQiM	Goal Question (Indicator) Metric
GQM	Goal Question Metric
H0	Null Hypothesis
H1	Alternative Hypothesis
HLD	High Level Design
HML	High Maturity Level
HP	Hewlett-Packard
ICSE	International Conference on Software Engineering
ID	Unique Identifier
IDEAL	Initiating Diagnosing Establishing
IEEE	Institute of Electrical and Electronics Engineers
IPM	Integrated Project Management (process area)
ISAM	Integrated Software Acquisition Metrics
ISO	International Organisation for Standardisation
IT	Information Technology
ITIL	Information Technology Infrastructure Library
KLOC	Thousand Lines of Code
KPI	Key Performance Indicators
LI	Largely Implemented
LML	Lower Maturity Level
LSR	Least Squares Regression
M2DM	Metamodel Driven Measurement
MA	Measurement and Analysis (process area)
MARE	Mean Absolute Relative Error
MER	Magnitude Error Relative
MIEIC	Integrated Master in Informatics Engineering and Computation
MinBU	Minimum number of Business Units
MK II FPA	Mark II Function Point Analysis
ML	Maturity Level
MLP	Multy-layer Perceptron
MMR	Multidimensional Measurement Repository
MMRE	Mean Magnitude Relative Error
MRE	Magnitude Relative Error
N/A	Not Applicable
NI	Not Implemented
NY	Not Yet
ODC	Orthogonal Defect Classification
ODM	Ontology Driven Measurement
OID	Organisational Innovation and Deployment (process area)
OMG	Object Management Group
OPM	Organisational Performance Management (process area, CMMI V1.3)
OPP	Organisational Process Performance (process area)
OT	Organisational Training (process area)
PA	Process Area

PB	Publication
PBC	Performance Benchmarking Consortium
PDCA	Plan Do Check Act
PDSA	Plan Do Study Act
PI	Performance Indicator
PIm	Partially Implemented
PMC	Project Monitoring and Control (process area)
PMI	Project Management Institute
PMP	Project Management Professional
PP	Project Planning (CMMI process area)
PPB	Process Performance Baseline
PPM	Process Performance Model
PPQA	Process and Product Quality Assurance (process area)
PRED	Percentage of Predictors
Price-S	Parametric Review Information for Costing and Evaluation – Software
PROBE	PROxy-Based Estimation Method
PSM	Practical Software Measurement
PSO	Particle Swam Optimization
PSP	Personal Software Process
QPM	Quantitative Project Management (process area)
QUASAR	Quantitative Approaches on Software Engineering and Reengineering
RBF	Radial Basis Function
RD	Requirements Development (process area)
REQ	Requirements
REQINSP	Requirements Inspection
REQM	Requirements Management (process area)
RQ	Research Question
RSK	Risk
RSKM	Risk Management (process area)
SAM	Supplier Agreement Management (process area)
SBO	Software Benchmarking Organisation
SCAMPI	Standard CMMI Appraisal Method for Process Improvement
SD	Standard Deviation
SDA	Survey Data Analysis, further analysis of a survey conducted by the SEI
SEER-SEM	Software Evaluation and Estimation Resources – Software Estimation Model
SEI	Software Engineering Institute
SEMA	Software Engineering Measurement and Analysis
SG	Specific Goal
SLIM	Software Lifecycle Management
SME	Small Medium Enterprise
SP	Specific Practice
SPI	Software Process Improvement
SPR	Software Productivity Research
SRS	Software Requirements Specification
SSIC	Systems and Software Consortium, Inc.
SVM	Support Vector Regression
TRW	Tandem Random Walk
TS	Technical Solution (process area)

TSP	Team Software Process
UCP	Use Case Points
UML	Unified Modelling Language
USA	United States of America
V	Version
VAR	Variance Account For
VARE	Variance Absolute Relative Error
WP	Work Product
WBS	Work Breakdown Structure

Definitions

Affirmations "Oral or written statement confirming or supporting implementation (or lack of implementation) of a model practice provided by the implementers of the practice, provided via an interactive forum in which the appraisal team has control over the interaction." (CMU/SEI, 2011c)

Artefacts "Tangible forms of objective evidence indicative of work being performed that represents either the primary output of a model practice or a consequence of implementing a model practice." (CMU/SEI, 2011c)

Benchmark To take a measurement against a reference point. Benchmarking is a process of comparing and measuring an organisation with the business leaders located anywhere (Kasunic, 2006). The acquired information helps the organisation to improve its performance.

Constellation "A constellation is a collection of CMMI components that are used to construct models, training materials, and appraisal related documents for an area of interest (e.g., development, acquisition, services)." (CMMI Product Team, 2010)

Self-directed Teams Teams whose members sense the project needs without being told, help whenever is necessary and "do whatever is needed to get the job done." (Humphrey, 2006)

Data Sufficiency Rules Coverage rules that determine how much evidence (Affirmations and Artefacts) needs to be provided in the SCAMPI A (Byrnes, 2011).

Fully Implemented Sufficient artefacts and or/affirmations are present and judged to be adequate to demonstrate practice implementation (CMU/SEI, 2011c). No weaknesses are noted.

Largely Implemented Sufficient artefacts and or/affirmations are present and judged to be adequate to demonstrate practice implementation (CMU/SEI, 2011c). One or more weaknesses are noted.

Not Implemented Some or all data required are absent or judged to be inadequate (CMU/SEI, 2011c). Data supplied does not support the conclusion that the practice is implemented. One or more weaknesses are noted.

Not Yet "The basic unit or support function has not yet reached the stage in the sequence of work, or point in time to have implemented the practice." (CMU/SEI, 2011c)

Organisational Scope A subset of the organisational unit that is determined by selecting support functions and basic units to supply data for the SCAMPI appraisal (CMU/SEI, 2011c).

Organisational Unit The part of the organisation that is subject of a SCAMPI appraisal and to which results will be generalised (CMU/SEI, 2011c).

Partially Implemented Some or all data required are absent or judged to be inadequate (CMU/SEI, 2011c). Some data are present to suggest some aspects of the practice are implemented. One or more weaknesses are noticed.
OR
Data supplied to the team conflict. One or more weaknesses are noted.

Sampling Factors Rule to select the organisation Basic Units into subgroups that determine the organisational scope to be target of the SCAMPI A (Byrnes, 2011). Are used to ensure adequate representation of the organisational unit.

Standard Processes Processes that the organisation statistically controls to assure that the organisation and projects quantitative objectives are achieved.

Subgroups Subset of the organisational unit defined by sampling factors, that are basic units with common attributes (CMU/SEI, 2011c).

Chapter 1

Introduction

When we open the book "CMMI (Capability Maturity Model Integration) for Development" (Chrissis et al., 2011) and read the preface, the models are presented as "collections of best practices that help organisations improve their processes" and the CMMI for development (DEV) "provides a comprehensive integrated set of guidelines to develop products and services". For years, several successful stories have been presented to the world, showing the benefits organisations achieved when using the model, going from improving the quality of the products and processes, to reducing schedule, costs (Herbsleb and Goldenson, 1996) and amount of rework. Consequently, processes become more predictable and customer satisfaction increases (Goldenson et al., 2004). The model is an improvement tool that can be implemented step by step, to improve a process area, evaluate capability or maturity, or simply improve selected practices. CMMI also guides organisations in building measurement capability to provide the information necessary to support management needs, as stated in the Measurement and Analysis process area (Chrissis et al., 2011). For adequate use, it is necessary to understand the model as a whole. In the staged representation the CMMI model is composed of 5 maturity levels, each of them achieved with the implementation of the specific and generic goals prescribed in the model in a current maturity level and all the precedent ones. To satisfy a goal the generic and specific practices, or acceptable alternatives to them, need to be fulfilled. Levels 4 and 5 are called high maturity levels. In these levels the organisations need to have knowledge on simulation, modelling and statistical analysis that support building process performance models that are relevant to indicate the status of objective and measurable organisation goals, and have process performance baselines to quantitatively control process/product execution. In maturity level 5 the organisations use their knowledge and capability of anticipating the behaviour of their standard sub-processes to support decisions regarding performance improvements or resolution of problems. This implies that decisions made are based on evidence of the success of the solutions.

There is plenty information, tips and cases of what makes CMMI work available in order to help organisations improve, but it is still their choice how they shape their processes to respond to their business needs and reflect their culture. Besides, a process may be defined but the real process is the one actually being executed. Even on strict sets of rules, the real process may still differ from the documented desired process. As each organisation has a choice of how to implement the model, use the practices and perform their work, there is high variability when comparing performance results. Therefore, there are unsuccessful cases and organisations achieving a maturity model but not performing accordingly. The Software Engineering Institute (SEI) and the United States Department of Defence (DoD) (Schaeffer, 2004) expressed concern with high maturity implementation as not all organisations understood it well, which reflected on their performance, and the release of V1.3 was intended to fix this problem (Campo, 2012).

CMMI Version 1.3 emphasises improvements on the organisations' performance, i.e. it clarifies that organisations need to focus processes on their business goals and carry out performance improvements to achieve those goals that are continuously improving. The Standard Appraisal Method for Process Improvement^{SM} (SCAMPI^{SM}) appraises the alignment of the organisation's processes, activities and results with the CMMI model but its objective is not to measure performance. To the best of our knowledge there is no tool to measure organisations' performance and evaluate it as a result of the quality of implementation of the CMMI practices and/or goals.

1.1 Research Scope

Given the afore mentioned facts, we conducted the research in this Ph.D. with the purpose of contributing to prior knowledge, overcoming some of the current limitations found, proposing a Framework to Evaluate the Quality of Implementation of Process Improvements (EQualPI), of which we validated part of its modules, and providing guidance to continue our work and for future research in this area. I chose to represent EQualPI as $= \pi$ as I consider that once the organisations implement the CMMI practices focusing on the performance outcomes and its benefits, they will achieve perfection, and for me π (pi) is a perfect number.

1.1.1 Problem Definition

The problem to solve is composed of three main aspects. One is that CMMI has a **high variability of performance** within the same level (Radice, 2000; Schreb, 2010). Schreb (2010) compiled the problems of CMMI: implementations that do not impact projects, wide range of solutions for the same practice not all leading to high performance, and highly variable approaches to implementation that may not lead to performance improvement. The model is not prescriptive, it only provides guidance and therefore the performance of the organisations implementing it depends on factors related to the business and teams, but also on the methods used to perform the work and quality of implementation of the model. In fact, as Peterson stated, the big issue is CMMI implementation (Schreb, 2010). Furthermore, we state that the quality of the implementation of the practices has an impact in performance indicators, related to the organisations' objectives.

Another aspect of the problem is the **quality of implementation** of the model. Some organisations have difficulties in the selection of the implementation methods, others simply copy the model as if it was a standard, leading to bad implementations. We consider that if the quality of implementation is good, the performance of the organisation using CMMI is improved.

Lastly, we consider that there is a need for a **performance evaluation method** that can help organisations to assess the quality of implementation of the practices and if they are actually improving their results or not. Even though version 1.3 of the CMMI model is more focused on the organisation performance, the objective of the SCAMPI is not to measure performance but appraise organisations' compliance with the model. Regarding this problem we consider that it should be possible to define metrics that measure the quality of the implementation of the CMMI model and measure the effects of process improvements. Having this capability would help to avoid implementation problems by early recognition of failures.

We synthesise the problem that we tackled in the following statement:

Not all organisations using CMMI achieve the best performance results, which could be achieved using a good implementation of the model. If a relationship can be established between methods used to implement a practice and the performance results of that practice, such relationship can be used in a framework to evaluate the quality of implementation of that practice.

1.1.2 Research Questions and Hypothesis

Our research questions (RQ) were the following:

RQ 1 - Why do some organisations not achieve the expected benefits when implementing CMMI?

RQ 2 - Why does SCAMPI not detect implementation problems, or does not address performance evaluation in all maturity levels?

RQ 3 - What additional recommendations can we provide to organisations to help them avoid problems when implementing CMMI?

RQ 4 - How can we evaluate the quality of implementation of the CMMI practices, ensuring that organisations fully attain their benefits and perform as expected?

RQ 5 - Is it possible to define metrics to evaluate the quality of implementation of CMMI practices focused on their effectiveness, efficiency and compliance?

RQ 6 - Can we determine the effects, expressed in a percentage, of uncontrollable factors in an evaluation metric?

Based on the problem statement, and the theory that there is a relationship between the quality of implementation of a CMMI practice and the quality of the outcome of the application of that practice. Based on the definitions of hypothesis given by Rogers (1966); Sarantakos (1993, page 1991) and Macleod Clark and Hockey (1981) we formulate ours as follows:

– *It is possible to objectively measure the quality of implementation of the CMMI practices by applying statistical methods, in the analysis of organisations' data, in order to evaluate process improvement initiatives and predict their impact on organisational performance.*

To demonstrate our hypothesis we embarked on a quest to model a quality indicator to measure the quality of implementation of the CMMI Project Planning Specific Practice (SP) 1.4 "Estimate Effort and Cost".

1.1.3 Research Objectives

This research had the following main objectives:

Objective 1 – Identify problems and difficulties in implementation of CMMI to help define the problem to tackle: considering the high variability of results that CMMI organisations present, evaluate the quality of implementation of practices based on quantitative methods.

Objective 2 – Develop and validate a framework to evaluate the quality of implementation of the CMMI practices.

Objective 3 – Demonstrate the evaluation of quality of implementation by building the performance indicator model to evaluate the particular case of the Project Planning process's Specific Practice 1.4 "Estimate Effort and Cost".

1.2 Research Approach

To answer our research questions and design our research methods we conducted literature reviews. The starting point was to identify the problems and difficulties in the implementation of CMMI practices, why did they occur, what were their causes and how could they be overcome. Furthermore, we analysed other researches contributions to help solve the problem, define the areas that required further research and also to base our solution definition on. Such approach allowed us to define the EQualPI framework base concepts, and develop its metamodel. We conducted a first case study to confirm and find other problems and recommendations that were added to the Framework as part of the Procedures Package.

The results of our first case study also gave us evidence that process improvements, even being well documented, required attention to ensure objective and quantitative analysis of their benefits. Therefore we defined the Process Improvements procedure. One of the issues we found in the first case study, was that requirements reviewers did not find the defects classification taxonomy in use, adequate for requirements defects. Therefore, all defects were classified as *documentation*. For that reason, we validated the Process Improvements procedure by developing a classification taxonomy specific for requirements defects and conducting a field experiment. We piloted the improvement with undergraduate and graduate students. The classification list was later adopted by an organisation which recognised its value.

During the definition and improvement of EQualPI we performed a second case study to further build and sustain the procedures and better define how the evaluation of the quality of implementation of processes should be defined. The case study confirmed the results of the first one: when implementing high maturity levels, organisations found gaps in lower maturity levels and had difficulties in finding solutions to implement the new processes. That was in line with the first motivation to start this Ph.D, understand the lack of results of some high maturity organisations, find a solution to objectively evaluate the quality of the processes in order to achieve the claimed improvement results and build performance indicator models that could be useful in such evaluation. Therefore, we analysed the reported results of organisations that achieved high maturity and the methods they used to build their process performance models and baselines and raised further questions that required further analysis of those surveys data. The results we achieve improved our list of recommendations and integrated the CMMI Implementation package and we did a final case study, to once again verify the identified problems were also found, the recommendations followed and further improve the CMMI Implementation procedure.

For the reason that we wanted to demonstrate the Framework in a Specific Practice, given the model extension (22 Process Areas with their respective Specific Goals and Practices) we selected the effort estimation Specific Practice. The rationale for choosing it was the importance of doing good estimates to execute the project building the right product, with the expected quality and respecting the plan. To validate the process area we used the Effort Estimation Accuracy variable and defined it as a function of controllable and uncontrollable factors. To build the model and therefore validate the EQualPI evaluation process we conducted a literature review on effort estimation processes, factors and models. We then developed EQualPI's Data Dictionary to define and collect the necessary data to evaluate the effort estimation process. The development of the Data Dictionary was also based on the analysis of TSP projects documentation and so was the Domain Model, which identifies the relations between the variables at different levels (development cycle, project, organisation). We intended to use data of organisations projects that were: systematically collected; of which we could have further details about the estimation process and projects context; and that were valid. We used the SEI data of TSP projects to build our EEA model and test our hypothesis. The data was collected by the SEI using the Data Dictionary; we extracted further information from the TSP database and used the Domain Model to define the aggregation of data needed to be able to define the performance indicator model.

1.2.1 Contributions

With our research we developed the EQualPI framework, defined its metamodel and architecture. The research approach and Framework overview in themselves are a contribution for researchers and practitioners, constituting a methodology for analysing and evaluating processes performance, based on the quality of their results and, in particular, the CMMI processes and levels.

We validated the EQualPI modules **CMMI Implementation**, **Process Improvement**, **Data Dictionary**, **Domain Model**, part of the **Evaluation** at projects aggregation level, using the **Process Performance Indicator Model** that we built to evaluate Effort Estimation Accuracy.

1.2.2 Beneficiaries

The development and demonstration of our Framework will provide organisations a tool that can help them to:

- Implement CMMI, by providing a pool of methods that can be adapted to implement the practices, and performance indicators to monitor them;

- Choose methods not only for their adequacy to context but for their performance, in terms of effectiveness and efficiency, when compared to others;
- Monitor process performance in order to act before problems occur;
- Anticipate impact of process changes on the performance indicators;
- Prioritise performance improvements;
- More accurately understand the origins of certain results.

The CMMI Institute will be able to assess whether there were actual performance improvements in a given organisation from one appraisal to the next. Researchers will benefit from the principles we established with EQualPI and with the Data Dictionary and Domain Model information to help them analyse TSP data.

1.3 Thesis Organisation

The remainder of this thesis is organised as follows:

In chapter 2 Fundamental Concepts, we present the concepts necessary to understand this research and the remaining chapters of the thesis.

Chapter 3 Background and Related Work provides the necessary information to delimit the problem and the contributes of other researchers to help solve some of the problem components.

We present our contribution to solve open points identified on prior research, in chapter 4 The EQualPI Framework, the core of our research, where we detail the framework to evaluate the quality of implementation of the CMMI practices and how organisations can use it.

In chapter 5 EQualPI Validation, we validate the framework we presented in the previous chapter.

In chapter 6 Conclusions we guide the reader as to how the framework is extended to other practices, indicate our achievements and their impact in the problem resolution, and define the boundaries of this research. We leave the research open and point to directions for future work that needs to be done in this area.

Chapter 2

Fundamental Concepts

Software Engineering is a discipline that appeared from the necessity of producing software applying engineering principles (van Vliet, 2007). The Institute of Electrical and Electronics Engineers (IEEE or I-triple-E) defines it as "the application of systematic, discipline, quantifiable approach to the development, operation, and maintenance of software; that is the application of engineering to software" (IEEE Std 610:1990). This concept is aligned with Humphrey (1988), who defined a **Software Engineering Process** as being the "total set of software engineering activities needed to transform user requirements into software". The process may include requirements specification, design, implementation, verification, installation, operational support, and documentation. Fuggetta (2000) extends the definition by stating that a software process is defined as a coherent set of policies, organisational structures, technologies and artefacts necessary to develop, deploy and maintain a software product.

If software engineering is a "quantifiable approach" it needs to be measurable, thus one must apply measurement. In Figure 2.1 we present the measurement components and the relations between them. We then clarify these concepts in the next section.

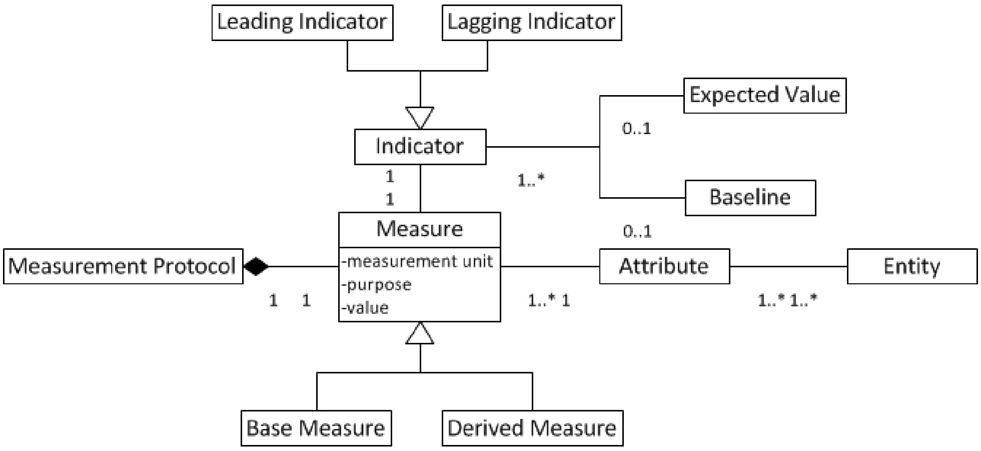


Figure 2.1: Measurement components. Based on ISO/IEC 15939:2007.

2.1 Measurement

There are several definitions for adequate terms to use in software engineering with respect to expressions such as measures, metrics, metrication, etc. (Zuse, 1997). Ragland (1995) clarifies the definitions of the terms "measure", "metric" and "indicator", based on the definitions of the IEEE and the Software Engineering Institute (SEI), and provides illustrative examples.

To **measure** (verb) (Ragland, 1995) is to ascertain or appraise to a standard that may be universal or local. It is considered an act or process of measuring; the result of measurement. The **measure** (noun) is the result of the act of measuring. An example of a measure is a single data point, for instance "today I produced 10 pages of my thesis". The data point to register would be *10*, i.e. the **value**. However, that information would be insufficient to analyse the measure. It is necessary to define the **measurement unit**, that in this case is *number of pages*, and the **purpose** of the measure, in this case *to know how long it took me to produce my thesis*. The measure refers to an **attribute**, "property or characteristic of an **entity** that can be distinguished quantitatively or qualitatively by human or automated means" (ISO/IEC 15939:2007). In ISO 14598:1998 a **metric** "is defined as a quantitative scale and method which can be used for measurement". Such term is often used to designate the data point and all information that allows collecting and analysing it, and that is the definition we will follow in this thesis. So the definition of metric (ISO/FDIS 9126-1:2000; ISO 14598:1998) is more suited to the **measurement protocol** that we define later in this section. A **base measure** is "defined in terms of an attribute and the method quantifying it (ISO/IEC 15939:2007)", measures a single property or characteristic of an attribute, which can be a product, process or resource (McGarry et al., 2002). Base measures are used to calculate **derived measures** (ISO/IEC 15939:2007) or, using the previous definition, metrics.

An **indicator** (ISO 14598:1998) is a measure that estimates or predicts another measure, which may be used to estimate quality attributes of the software or attributes of the development process. Ragland (1995) refers to indicator as a device or variable that is set to describe the state of a process, based on its results, or occurrence of a predefined condition. The indicator is an imprecise indirect measure of attributes (ISO 14598:1998) that provides insight into the software development processes and improvements concerning attaining a goal, by comparing a metric with a baseline or expected result (Ragland, 1995). A **performance indicator** is a measure that provides an estimate or evaluation of an attribute. The performance indicator is derived from a base measure or other derived measures. This means that a performance indicator can even be derived from other performance indicators. A **leading indicator** anticipates quality, as it is a measure that allows forecasting and diagnosis (Ferguson, 2008; Investopedia, 2007). On the other hand, a **lagging indicator** follows an event or tendency, therefore allows appraising (Investopedia, 2007).

Regardless of the scientific field, **measurement** (Pfleeger et al., 1997) generates quantitative descriptions of key processes, products and resources, those measures are useful to understand the behaviour of what is being measured. The enhanced understanding of processes, products and resources is useful to better select techniques and tools to control and improve them. Pfleeger et al. (1997) consider that software measurement exists since the first compiler counted the number of lines in a program listing. In 1971, Knuth reported on using measurement data, instead of theory, to optimise FORTRAN compilers, based on natural language. In the CMMI for development model constellation[1] (CMMI-DEV), the **process measurement** is considered to be a set of definitions, methods and activities used to take measurements of a process and the corresponding products for the purpose of characterising and understanding it (Chrissis et al., 2011).

One of the requirements for establishing a process measurement program is to put in place a measurement system. Kueng (2000) mentions two important characteristics of a process measurement system: it shall focus on the processes and not on the entire organisation or on organisation units, and the measurement system shall evaluate performance holistically, by measuring quantitative and qualitative aspects. Kueng was focused on business processes without considering the relevance of measuring products characteristics and the importance of analysing processes results at different levels (projects, business units and organisations), for a complete measurement system. Kitchenham et al. (1995) enunciate some of the necessary concepts to develop a validation framework to help researchers and practitioners to understand:

- How to validate a measure;
- How to assess the validation work of other people;
- When it is appropriate to apply a measure according to the situation.

In the same work, Kitchenham et al. (1995) define **measurement protocols** as necessary elements to allow the measurement of an attribute repeatedly and consistently. These characteristics contribute to the independence of the measures from the measurer and the environment. The measurement protocol depends on how the measured value is obtained and on the use that will be given to the measure. A measure is therefore applied to a specific attribute on a specific entity using a specific measurement unit for a specific purpose.

Doing proper measurement and analysis is fundamental to evaluate processes and products development performance, and to improve them. Process measurement allows inferring the performance of the processes.

2.2 Continuous Process Improvement

There are several continuous process improvement frameworks, some of which we present in this subsection: Shewhart Cylce PDSA (Plan, Do, Study, Act) - derived from the more known PDCA cycle (Plan, Do, Check, Act), Deming's wheel (Moen and Norman, 2006), Juran's Quality Improvement Process (Juran and Godfrey, 1998), Six Sigma's DMAIC (Define, Measure, Analyse, Improve and Control) (Hahn et al., 1999) and the IDEAL model (Initiating, Diagnosing, Establishing, Acting and Learning) (McFeeley, 1996).

The Japanese defined the PDCA cycle naming it the Deming Wheel in 1951, which was already based on a refined 4 steps product improvement Deming presented in 1950 (added a 4th step

[1] The definition of constellation can be found in Acronyms and Definitions.

- "redesign through marketing research") of the 3 steps cycle defined by Shewhart in 1939 (Specification, Production, Inspection) (Moen and Norman, 2006). In 1993, Deming named PDSA the Shewhart cycle for learning and improvement, which includes the following steps (Figure 2.2):

- Plan: plan change or test, aimed at improvement;
- Do: carry out the change or test;
- Study: analyse results to gather lessons learnt or understand what went wrong;
- Act: adopt the change or abandon it, alternatively run through the cycle again.

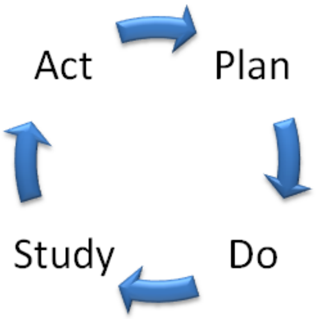


Figure 2.2: Shewhart cycle for learning and improvement: Plan, Do, Study, Act (Moen and Norman, 2006).

The quality improvement process (Juran and Godfrey, 1998) is defined as a continuous process established in the organisation, with a governance model and intended to last through the organisation lifetime involving the definition of a plan, roles and upper management roles. The author considered the relevance given to new developments was more emphasised and structured than the necessity of reduce "chronic waste", giving less attention to quality. The details of organising and forming a quality council are described, as well as those of preparing the improvement projects including criteria and roles needed in the improvement project team. We emphasise the steps needed for an improvement, summarised as below:

- Awareness: have proof of the need;
- Determine the potential return on investment;
- Select processes and project to implement;
- Diagnosis journey: understand the symptoms;
- Formulate theories to identify the causes and select the ones to be tested, do retrospective analysis and check lessons learnt;
- Remedial Journey: choice of remedies to remove the causes;

- Establish controls to hold the gains;
- Institutionalise the process improvement.

Six Sigma was used in Motorola to reduce defects but was published and used by many others. It has an approach for improving and also eliminating defects, DMAIC (Hahn et al., 1999) (see Figure 2.3):

- Define: the problem, its impact and potential benefits;
- Measure: identify the relevant measurable characteristics of the process, service or product, define the current baseline and set the improvement goal;
- Analyse: identify the process variables causing the defects/inefficiencies;
- Improve: establish acceptable values for those variables and improve the process to perform within the limits of variation;
- Control: continue measuring the new process to ensure the process variables have the expected behaviour and achieve the established improvement goal.

Define
Measure
Analyse
Improve
Control

Figure 2.3: Six Sigma's DMAIC: Define, Measure, Analyse, Improve, Control (Hahn et al., 1999).

The IDEAL model is a continuous process improvement framework published by the SEI (McFeeley, 1996) consisting of five phases and their respective activities (Figure 2.4):

- Initiating: after a stimulus for the improvement, this phase includes activities to characterise the need and start a project. Includes the activities of: set context, build sponsorship and charter infrastructure;
- Diagnosing: understanding the current state (*as is*) and defining what the desired future state (*to be*) is. Includes the activities of: characterise current and desired states, set priorities and develop approach;
- Establishing: definition of the plan on how to achieve the desired state and test and improve the solution to implement. Includes the activities of: plan actions, create solution, pilot test solution, refine solution and implement solution;
- Learning: analyse what was done, understand if the goals were achieved, learn from the experience and prepare for future improvements. Includes the activities of: analyse and validate and propose future actions.

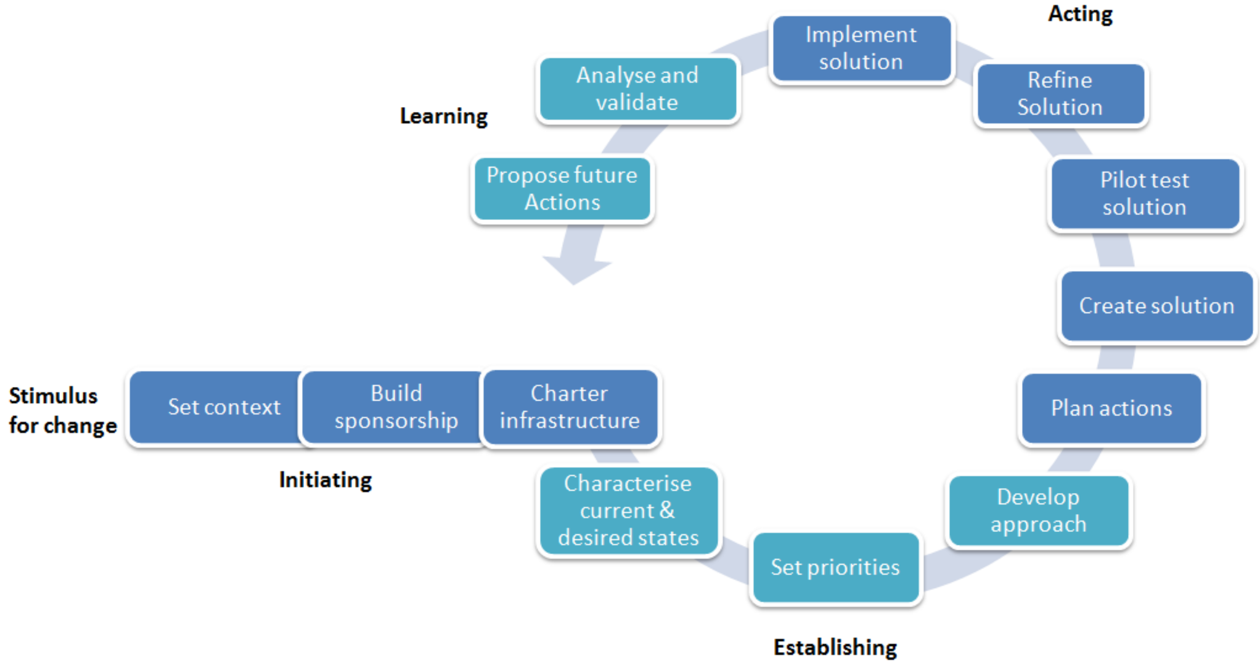


Figure 2.4: IDEAL model: Initiating, Diagnosing, Acting, Learning (Gremba and Myers, 1997).

2.3 Process Performance Measurement and Improvement

CMMI-DEV defines **process performance** as a measure of the actual results achieved by following a process (Chrissis et al., 2011). It is characterised by both process measures and product measures. The process performance models depend on historical data of the processes performance and on which data is collected by the measurement system in place.

According to CMMI-DEV (Chrissis et al., 2011), the **process performance model** (PPM) describes the relationships amongst the attributes and the work products of a process. The relationships are established from historical data of the process performance, and the calibration of the model is done using data collected from the product, process and metrics of a project. Consequently, the process performance models are used to predict the results achieved by following the process that the model represents. Kitchenham et al. (1995) indicate that in predictive models, such as COCOMO (COnstructive COst MOdel), variability of the predicted values may occur, caused by the incompleteness of the model, as there are factors that affect what is being predicted which may not have been considered in the model. The model error is the sum of the model incompleteness and the measurement error.

When the data collected on measures is stable, and the process performance model adequately supports the prediction of the behaviour of the projects, it is possible to understand the normal behaviour of the process, i.e., under known circumstances. The **process performance baseline** (PPB) characterises the behaviour of the process by establishing the maximum and minimum values where the process behaves under the expected causes of variation (Florac et al., 2000). If a project or process behaves outside boundaries, by a certain threshold, the model shall allow the anticipation of that occurrence. In that case the team needs to identify the special causes of the variation. If the variation brings negative consequences then the team needs to act in order to prevent deviation of the project or process. CMMI-DEV (Chrissis et al., 2011) defines process performance baseline as a documented characterisation of the actual results achieved by following a process. The baseline is used as a benchmark[2] to compare the actual performance of the process with its expected performance.

In his doctoral thesis, Dybå (2001) indicates that a broad definition of **Software Process Improvement** (SPI) would include the following activities:

- Define and model a software process;
- Assess the process;
- Refine the process;
- Innovate by introducing a new process.

Our perception is that to achieve process improvement it is necessary to measure the initial performance in order to compare it with the final performance. The objective of the process improvement, after all, is to improve the process performance and we do not want organisations to loose the focus on that goal. For that reason, we introduce the term **Software Process Performance Improvement**. So Dybå's activities are updated here to include the term performance:

- Define and model a software process **performance**;
- Assess the process **performance**;
- Refine the process;
- Innovate by introducing a new process **or new process version**.

Considering that a process improvement must be measured and of value, it has to result in performance gains, hence a process improvement should not be done without considering that the process performance must be improved. Some may argue that a process improvement *per se* implies a performance improvement. Nonetheless, many organisations implement "improvements" without measuring the performance of the process *as is* and the final performance. Moreover, even if a particular process improvement leads to its better performance it may have a negative impact in other processes that cannot be perceived if the organisation does not do an overall control. In fact, an improvement of a process may coincide with a process improvement project but may result from another process change that may have been planned or not. For these reasons we consider that it is important to align process improvements with the organisation goals and focus on the correct outcome.

[2] For a definition of benchmark please refer to Acronyms and Definitions.

The task of identifying performance indicators which can show how the process is performing is not trivial. First of all, performance indicators *per se* do not necessarily show if the organisation is doing better or worse. Those indicators need to be related with the organisation business objectives. That is what it makes metrics implementation a challenge. To make the task easier, organisations can use tools such as the Goal Question (Indicator) Metric, Balanced Score Card (BSC) or the Goal-driven Measurement (Park et al., 1996), a combined application of the BSC and Goal Question (Indicator) Metric (GQiM). We illustrate the method in Figure 2.5.

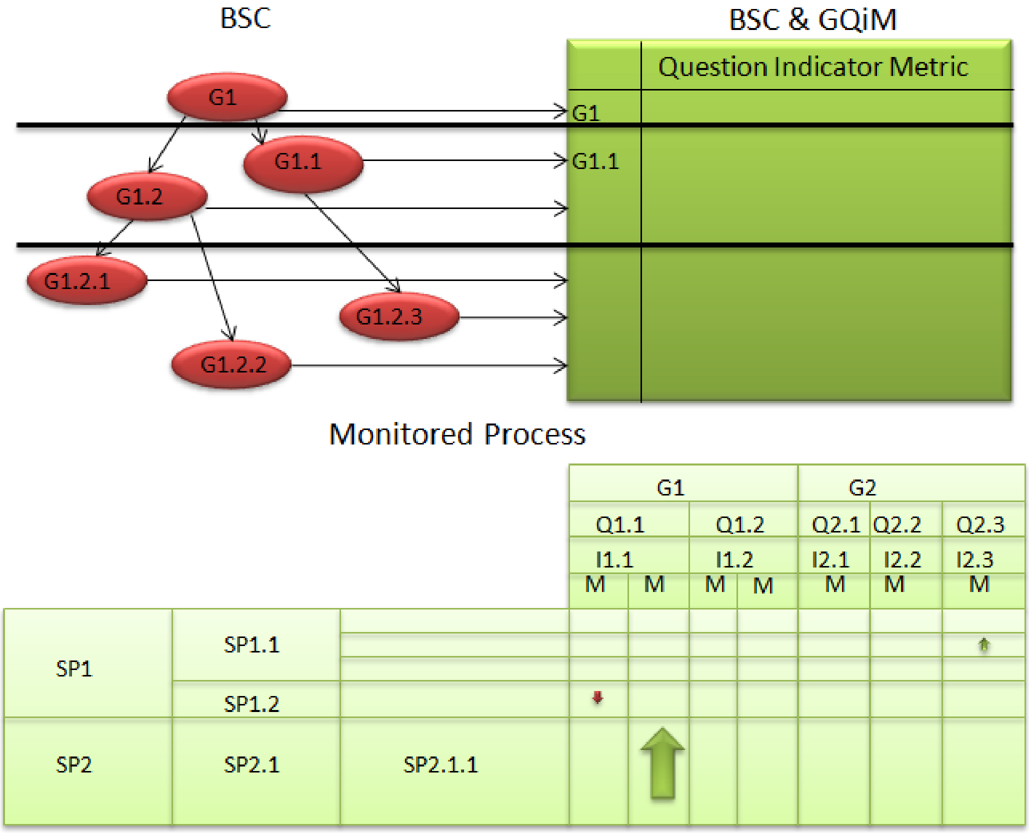


Figure 2.5: Mapping BSC into GQiM, into processes and sub-processes(SP). Monitored processes are being followed.

Note: Realistic metric (M) goals (G) are established, which may be to decrease a metric value, such as number of defects; or increase a metric value, as % of code being reviewed.

The business goals metrics are established using the BSC and are drilled down from organisation's goals to business units' goals and ultimately projects' and individuals' goals. The metrics are derived by using the GQiM and mapped with the organisations different levels of goals. The most relevant goals/sub-goals for the business strategy are elicited, and realistic objective targets are established for those indicators goals. The metrics to determine the indicators are collected in different sub-processes. If the current processes performance does not allow achieving the quantitative goals, then process performance improvement projects can be conducted in order to find solutions to achieve them.

Process performance improvements result in updates in Process Performance Baselines and eventually in Process Performance Models. The Process Performance Baselines help defining Process Performance Models and there are bidirectional relationships between what needs measurement and what builds measurement (processes, performance, models, baselines and improvements). CMMI has practices of Measurement and Analysis, at maturity level 2, and practices to build process performance models and establish process performance baselines in the the Organisational Process Performance process area (PA), at maturity level 4.

2.4 CMMI Architecture and Appraisal Method

CMMI has two representations designated continuous and staged (Chrissis et al., 2011). The **continuous** representation is organised in Capability Levels (CL), going from 0 to 3, while the **staged** representation is organised in Maturity Levels (ML) that range from 1 to 5. In our research we refer to the staged representation, because CLs are just applied to individual process areas, whereas MLs are applied across **Process Areas**. However, the framework we developed is usable in both representations as one organisation may select just a process area to evaluate or do a broader evaluation. In Figure 2.6 we present the CMMI maturity levels.

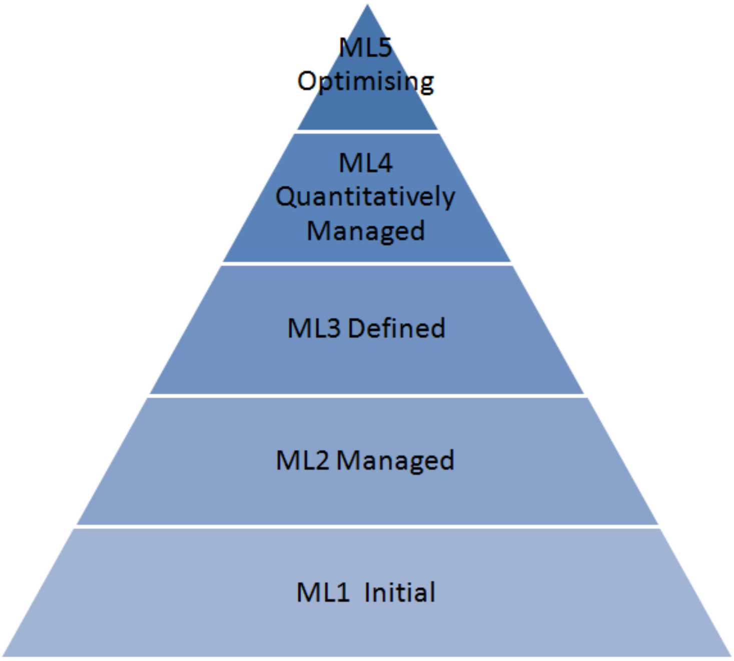


Figure 2.6: CMMI maturity levels in the staged representation.

To achieve a ML it is necessary to accomplish the Specific and Generic Goals of that ML and the precedent ones. In the **Initial** level (ML 1) there are no formal processes (Chrissis et al., 2011).

In ML 2, **Managed**, the processes are planned and executed according to the organisation's policy. The projects have documented plans necessary for their management and execution. At this ML, the process description and procedures can be specific to a project. The statuses of the projects are visible to management and commitments with relevant stakeholders are established and revised as needed.

In ML 3, **Defined**, the standard processes are used to establish consistency across the organisation and are continuously established and improved. The procedures used in a project are tailored from the organisation set of standard processes. The interrelationships of process activities and detailed measures of processes, work products and services are used to manage processes.

At ML 4, **Quantitatively Managed**, the organisation establishes quantitative objectives for quality and process performance. The projects have quantitative objectives, based on the goals of the organisation, customers, end-users and process implementers expectations. The projects' selected sub-processes are quantitatively managed, i.e., the data of specific measures of the process performance are collected and analysed. The process performance baselines and models are developed by setting the process performance objectives necessary to achieve the business goals. The processes' performance becomes predictable, based on the projects' and processes' historical data.

At ML 5, **Optimising**, the quantitative understanding of the business objectives and performance supports the organisation improvement decisions. The defined and standard processes' performance, the supporting technology, innovations and business objectives are continuously improved based on the revision of the organisational performance and business objectives. The improvements are quantitatively managed. Maturity Levels 4 and 5 are known as High Maturity Levels (HMLs).

In CMMI the process areas are organised in categories, namely **Process Management**, **Project Management**, **Engineering** and **Support** (Chrissis et al., 2011). The Process Areas include **Specific Goals** to accomplish, each of them presenting **Specific Practices** that help achieve those goals. Besides, at levels 2 and 3 the model includes **Generic Goals**, with the respective **Generic Practices**, applicable across process areas. When organisations are appraised at a ML or CL, the analysis is focused on what the organisations practices are to achieve the Specific Goals within that level.

SCAMPI is the method used to benchmark the maturity of a company in terms of the CMMI model (CMU/SEI, 2011c). This method is used to identify strengths and weaknesses of the processes and determine the company's capability and maturity level. There are three SCAMPI classes: A, B, and C. Class A is the most formal one, and is required to achieve a rating for public record. The other two apply when companies are implementing internal improvements at lower costs. In the remainder of this section we present two groups of SCAMPI rules that we believe should be considered in the evaluation of CMMI implementations, i.e., not the rules focused on planning the SCAMPI, but the **sampling factors** and **data sufficiency** rules[3]. Knowing how organisations are rated at a CMMI level and the SCAMPI rules is important to help answer the research question **R2** - *Why does SCAMPI not detect implementation problems, or does not address performance evaluation in all maturity levels?*

[3] The definitions of these terms can be found in Acronyms and Definitions.

It is not cost and effort effective to appraise an entire organisation and all its projects. It is therefore important to have sampling rules that ensure that the subset of the organisation and projects that are appraised are representative of the overall organisation. Sampling organisation units is done by following the steps (CMU/SEI, 2011c; Byrnes, 2011):

Sample Rule 1 - Understand the organisation unit and how it is organised. The organisation unit is composed of basic units and support functions;

Sample Rule 2 - Determine the organisation unit process drivers that influence how the processes are implemented;

Sample Rule 3 - Organise basic units and support functions into subgroups by applying sampling factors (e.g. location, customer, size, organisational structure and type of work);

Sample Rule 4 - Use equation 2.1 to determine the minimum representative sample that is collected from the subgroups and are included in the organisational scope.

A good principle is to evaluate elements of the organisation in proportion to their contribution:

$$MinimumNumberofBasicUnits: \quad MinBUn = \frac{Subgroups \times BUn}{TotalBUn} \tag{2.1}$$

where **MinBUn** is the minimum number of basic units to be selected from a given subgroup, **Subgroups** is the number of subgroups, **BUn** is the number of basic units in the given subgroup and **Total BUn** is the total number of basic units. When the computed value is less than 1 the required number of BUn is 1, when is greater than one the number of BUn is given by rounding the number to 0 decimal places.

The coverage rules (CMU/SEI, 2011c; Byrnes, 2011) determine how much should be collected in the appraisal:

Coverage Rule 1 – Each basic unit or support function sampled must address all practices in the process areas for which they supply data.

Coverage Rule 2 – For each subgroup at least one basic unit shall provide both artefacts and affirmations. The sampled basic unit shall provide data for all process areas.

Coverage Rule 3 – For at least 50 percent of the basic units within each subgroup, both artefacts and affirmations shall be provided for at least one process area.

Coverage Rule 4 – For all sampled basic units in each subgroup either artefacts or affirmations shall be provided for at least one process area.

Coverage Rule 5 – Both artefacts and affirmations shall be provided for each support function for all process areas relating to the work performed by that support function.

Coverage Rule 6 – The artefacts and affirmations provided by a support function shall demonstrate the work performed for at least one basic unit in each subgroup.

Coverage Rule 7 – In cases where multiple support functions exist within the organisational unit, all instances of the support function shall be included in the appraisal scope.

After all evidence is collected, the appraisal team characterises the implementation of CMMI practices for each model practice and each basic unit or support function. The practices implementation is classified as a **weakness** or a **strength**. Based on this classification each practice in each basic unit or support function is characterised as **Fully Implemented** (FI), **Largely Implemented** (LI), **Partially Implemented** (PIm), **Not Implemented** (NI) or **Not Yet** (NY)[4]. The rules for aggregating implementation-level characterisations to derive organisational unit-level characterisation are summarized in Table 2.1.

Table 2.1: Rules to aggregate implementation-level characterisations (CMU/SEI, 2011c).

Characterisation	**Implementation**
Fully Implemented (FI)	All FI or NY, with at least one FI
Largely Implemented (LI)	All LI or FI or NY, with at least one LI
Largely or Partially Implemented (LI or PIm)	At least one LI or FI and at least one PIm or NI
Partially Implemented (PIm)	All PIm or NI or NY, with at least one PIm
Not Implemented (NI)	All NI or NY, with at least one NI
Not Yet (NY)	All NY

If any practice is not characterised as FI it is necessary to explain the gap between the organisation practice and what the model expects (CMU/SEI, 2011c). A weakness, "ineffective, or lack of, implementation of one or more reference model practices", is only documented if it has impact on the goal. A goal is rated **Satisfied** if, and only if, all associated practices at organisational unit level are characterised as LI or FI and the aggregation of weaknesses of the goal do not have negative impact on its achievement.

CMMI establishes quality principles – what to do. It presents guidelines in terms of a set of good practices to achieve goals that also concern the organisation as a whole, but does not define the processes. The Team Software Process (TSP) is used together with the Personal Software Process (PSP) providing organisations disciplined processes to be used by individuals and teams (Davis and McHale, 2003). PSP was defined by Humphrey himself while developing 60 software programs applying all SW-CMM (Software Capability Maturity Model) practices up to level 5. While PSP is used by individuals TSP helps them work together as self-managed teams. TSP shows how to do things by providing the necessary steps to do the work and forms to register plans and work execution information.

[4] The definitions of these terms can be found in Acronyms and Definitions.

2.5 TSP Architecture and Certification

TSP includes process scripts to achieve high maturity performance that are to be followed by self-directed teams, i.e., teams make the decisions together instead of having them imposed by a leader or manager. In such teams anyone can assume the roles necessary for project completion; there is a team leader, who is part of the team, and a coach, who is an observer helping individuals and team to be on track. "All team members participate in planning, managing and tracking their own work" which is "key for high motivation and high performance in knowledge-based work" Faria (2009). The team members shall receive training in Personal Software Process (PSP), so that individual team member's skills are built.

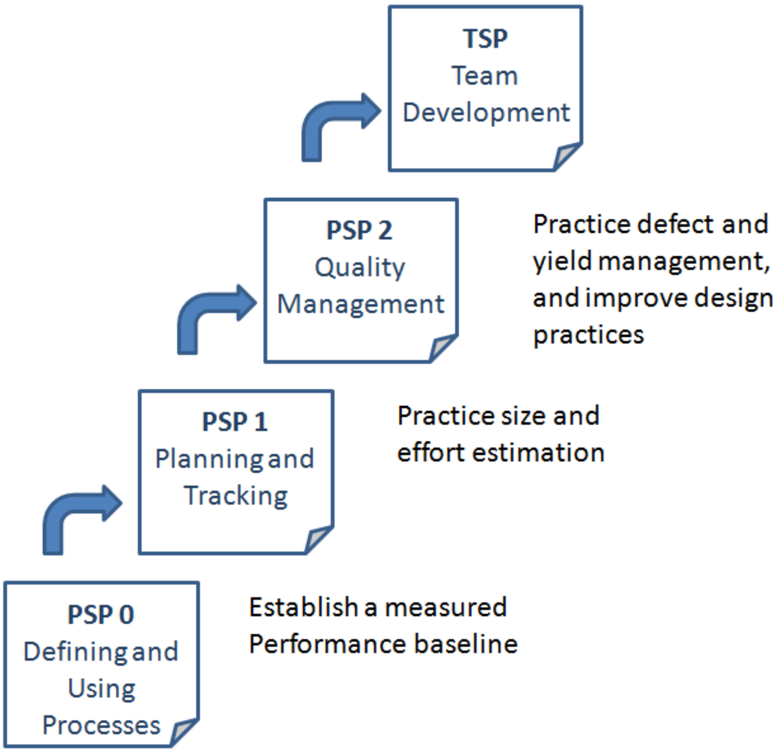


Figure 2.7: Personal Software Process (PSP) training (Faria, 2009).

Note: PSP training is introduced stepwise in a sequence of small projects; people get convinced by seeing their performance improved with practice. The last step is Team Software Process (TSP^{SM}) training.

PSP training is a path for self-improvement and discipline, consisting of developing several products and improving the process by adding quality practices (Lopes Margarido, 2013). The training is done in steps (Figure 2.7) beginning with **PSP0**, where developers write their development process and follow it. During development, programmers record information about the program: size, development time, and number of defects (Davis and McHale, 2003). At the beginning of **PSP1** programmers use the data collected in the previous phases to plan their work and to estimate the necessary effort to develop the program and its size. Such data is used in **PSP2** to alert programmers to their mistakes, to when they are made and to the quality practices to avoid them. The programmers plan for expected numbers of defects inserted and removed in each development phase, and hence learn to manage defects and yield. PSP training towards expertise goes from simple and unplanned to complex and predictable (Lopes Margarido, 2013). We find that "PSP is the exact application of engineering to software (Lopes Margarido, 2013)" and, as van Vliet (2007) stated "discipline is one of the keys" to successfully complete a development project. The benefit of PSP is that programmers see their results improving during training; at its end their skills and programs are unequivocally better. The processes are embraced, not imposed. Furthermore, programmers become aware of their personal data, allowing them to set individual goals for improving their own development process towards making better products, faster.

The introduction of TSP in an organisation is done gradually, in two phases, the **Pilot** and the **Roll-out** (Faria, 2009). The pilot phase includes training a controlled small number of project teams and their managers, launch the teams with TSP followed by a TSP coach, execute the project using TSP and gather data that will support results evaluation at the end of the project. The roll-out phase requires the training and certification of internal TSP trainers and coaches to gradually launch additional teams at a sustainable pace. As people integrate new teams, they bring in their knowledge and experience using TSP.

The basis of TSP teams is individual team members that have built their skills and completed PSP training. The team is built in the project launch week, by setting goals, assigning roles, tailoring the team's processes and designing detailed balanced plans with the active participation of all team members. Team management is done by managing communication and coordination, tracking the project and analysing risks. The project is developed and planned iteratively, it has a launch and a post-mortem meeting and is done in cycles, each of them beginning with a launch/relaunch[5] meeting and finishing with a post-mortem meeting. This iterative development should be based on the most adequate development model, which is determined by the technical and business context of the project (Faria, 2009). It can be done in small iterations, using a spiral model with increasing functionalities and/or complexity, or it can be sequential, following a waterfall model. The status of the project is reviewed weekly.

A TSP project includes different phases:

- **Planning**, done in the launch and relaunch meetings, in which all team members participate, so everyone's commitment is assured. Roles are assigned, the processes are selected, size of work products and effort to do tasks are estimated;

- **Development**, not only of the Code itself but eliciting Requirements, doing High-Level Design and Detailed Design;

- **Defect Removal**, which includes PSP processes of personal review, unit testing and compile, and also inspections, peer-reviews, integration and system testing.

[5] In case of not being the first launch meeting.

TSP and PSP require collecting "four core measures, which are the basis for quantitative project and quality management and project improvement, at personal, team and organisation levels Faria (2009)". The actual and planned values of size, effort, quality and schedule are recorded and controlled.

TSP-PACE (or just PACE), TSP Performance and Capability Evaluation, is the process to evaluate the TSP data of software development organisations and programs towards the TSP certification of organisations (Nichols et al., 2013). The program certification involves assessing the quality of the following TSP elements:

- Data;
- Training;
- Coaching;
- Launches and relaunches;
- Post-launch coaching;
- Project cycle post-mortem reports;
- Project post-mortem reports.

For organisational certification the teams under the certification scope are assessed in the aforementioned dimensions and the organisation is also assessed in **scope**, **quality and quantity of the gathered data**, and **customer satisfaction data**, regarding the work done by the TSP teams. All data are analysed in the five dimensions that constitute the profile: coverage, process fidelity, performance, costumer satisfaction and overall. Each profile has multiple variables that are evaluated.

2.6 Effort Estimation in CMMI and TSP

The CMMI Project Management category includes the Project Planning (PP) Process Area (Chrissis et al., 2011). Project Planning is amongst the first activities necessary to start a software development project and plays an important role in the course of the project. The plan can be revisited whenever necessary, being it because the process used is done in cycles, at the beginning of which detailed estimates are provided for the necessary tasks to execute them, as in Scrum (Schwaber and Sutherland, 2016) or TSP (Humphrey, 2006), or just because the scope changed and it is necessary to re-plan to accommodate the necessary activities. Therefore, project planning plays an important role not only on the execution of the project but also on the quality of the outcomes. As a CMMI process area, Project Planning is part of ML 2 with the purpose of establishing and maintaining plans that define the project activities (Chrissis et al., 2011). We include the specific goals and practices summary of this Process Area in Table 2.2, to show what goals are expected to be achieved when planning projects following CMMI, in order to be able to develop the project plan, involve relevant stakeholders, have team commitment to the plan and maintain the plan.

Table 2.2: Specific Goals and Practices of Project Planning Process Area (Chrissis et al., 2011).

SG 1	Establish Estimates
SP 1.1	Estimate the Scope of the Project
SP 1.2	Establish Estimates of Work Product (WP) and Task Attributes
SP 1.3	Define Project Lifecycle Phases
SP 1.4	Estimate Effort and Cost
SG 2	Develop a Project Plan
SP 2.1	Establish the Budget and Schedule
SP 2.2	Identify Project Risks
SP 2.3	Plan Data Management
SP 2.4	Plan the Project's Resources
SP 2.5	Plan Needed Knowledge and Skills
SP 2.6	Plan Stakeholder Involvement
SP 2.7	Establish the Project Plan
SG 3	Obtain Commitment to the Plan
SP 3.1	Review Plans That Affect the Project
SP 3.2	Reconcile Work and Resource Levels
SP 3.3	Obtain Plan Commitment

Focusing on the estimation process, according to CMMI (Chrissis et al., 2011) in order to establish the estimates of the project's planning parameters and achieve Specific Goal 1 (SG1): the parameters need to be identified, have a sound basis and consider project's requirements from relevant stakeholders; the estimates of the parameters must be documented along with the information sustaining them; and it is necessary to have the team's commitment to those estimates. The SG can be accomplished by following Specific Practice 1.1 (SP1.1) to SP1.4. All the estimation is done under the project scope (SP1.1), that is broken down, and depends both on the estimates of the Work Product (WP) and task attributes (SP1.2), and on the definition of the project lifecycle phases (SP1.3). Therefore, SP1.4 "Estimate Effort and Cost" is done using the prior SPs as inputs.

TSP provides a project planning framework compliant with the CMMI and a set of planning guidelines to support planning (Humphrey, 2006). Ideally, the teams use their historical TSP data, but if unavailable the values in the guidelines can be used initially. In case the data are not adequate for that project/team case it is recommended they use their best estimate. TSP teams will rapidly get their own data to estimate next because they will be gathering data as they work and feeding their historical database, rendering such an initial estimation relative.

There are several TSP plans produced. The **overall plan** is produced by the team and is adjusted later once the next phase balanced plan is done. The **bottom-up plan** is done considering task decomposition for the next phase and is done by each team member individually. That plan is then load balanced to consider the individual plans and produce the team's **balanced plan** for the next phase.

TSP does not provide initial values to estimate Requirements, Requirements Inspections and High-Level Design. Nonetheless, guidelines are provided in the Quality Plan (Humphrey, 2006, pages 148, 149) to define the time balance between inspection and development of requirements and high level design. For example, the guideline for the ratio between detailed design and coding time is that it should be higher than 1.

To help teams planning the implementation phase, TSP provides guidelines for the development rate per hour of the total code, which requires estimating the product/component size, percentage of time spent in phase (Humphrey, 2006, page 131), and the ratio between defect removal and corresponding defect insertion phase. The implementation phases are Detailed Design, Detailed Design Review, Detailed Design Inspection, Coding, Code Review, Compiling, Code Inspection and Unit Test.

In TSP the method used to produce the estimates of size and time is the one used on PSP, namely the PROxy-Based Estimation (PROBE) (Humphrey, 2005). The method is based on the definition of any item that can be used as a proxy. It requires that the team has the capability to provide an initial draft of the detailed design that allows them to estimate the size (added, modified, removed and actual) of the parts composing the solution to develop. When there is no historical data, expert judgement is used to estimate size and time, PROBE D. When there is some data that can be used, size is estimated based on it and so is time, PROBE C. PROBE B uses historical data to estimate size and time and a regression model with correlation equal or higher than 70%. While PROBE A, the desired method, uses historical data of the proxy to do the estimation using a linear regression model with correlation equal or higher than 70%.

Chapter 3

Background and Related Work

This chapter presents the results of the literature review done to define the problem, understand the previous contributions of other researchers, and identify the open ends of the problem that still need to be addressed. This is needed to understand the underlying basis of our work and the areas to which other researchers already contributed.

3.1 Process Improvements

In this section we discuss the evolution of metrics programmes and the CMMI model, diverted to the problems that persist. We then analyse which frameworks exist to help solve the problem and indicate the limitations that remain. By the end of the section the motivation for our research and the problem we tackled should be clear.

3.1.1 Historical Perspective on Metrics Programs and CMMI

In 1978 a group of Hewlett-Packard (HP) engineers went to Japan to study the techniques used in manufacturing that led to significant improvements (Grady, 1992). Those techniques were analysed to investigate how they could be applied to the HP's software development processes. The HP experience is a good example of the implementation of high maturity practices even before the idealisation of the CMMI level 5 (Consultant, 2009, personal communication).

According to Jones (1991), since 1979 it has been possible to have stable metrics and accurate applied measurement of software. Organisations such as IBM, Hewlett-Packard, Tandem, UNISYS, Wang and DEC, measured productivity and quality and used data to make planned improvements. Du Pont, General Electric and Motorola were innovative in software measurements. ITT, AT&T, GTE and Northern Telecom were pioneers in quality and reliability measures. Several management companies such as Software Productivity Research; DMR Group, Peat, Marwick & Mitchell, Nolan, Norton & Company, and Ernst and Young, were more effective than universities in using metrics and in transferring the technologies of measurement throughout their client base.

Phil Crosby assembled a maturity framework in 1979 and at the beginning of the 1980's IBM began the assessment of the capabilities of many of its development laboratories. Later, the SEI incorporated Deming[1] principles and Shewhart[2] concepts of process management, published by Deming in 1982 (Humphrey, 1992). Crosby's framework originated an assessment process and generalised maturity framework that was published by Radice in 1985. In April 1986, Watts S. Humphrey stated in the IEEE Spectrum that complex systems could be programmed with high quality and reliability if done by strong technical teams using a highly disciplined software process (Callison and MacDonald, 2009). The concepts of the SEI and MITRE Corporation were compiled into a technical report by Humphrey that was published in 1987 (Humphrey, 1992). In 1989, Humphrey added that the problems in software engineering are not technological but have a managerial nature (Dybå, 2001). The Personal/Team Software Process were released in 1990 (Callison and MacDonald, 2009) and in 1991 Mark Paulk clarified the content of the maturity framework with the publication of the Capability Maturity Model (CMM) by the SEI (Humphrey, 1992). Since its creation until now the CMM has evolved, giving way to the CMMI, now in version (V) 1.3. The evolution of the model is represented in the chronological diagram in Figure 3.1.

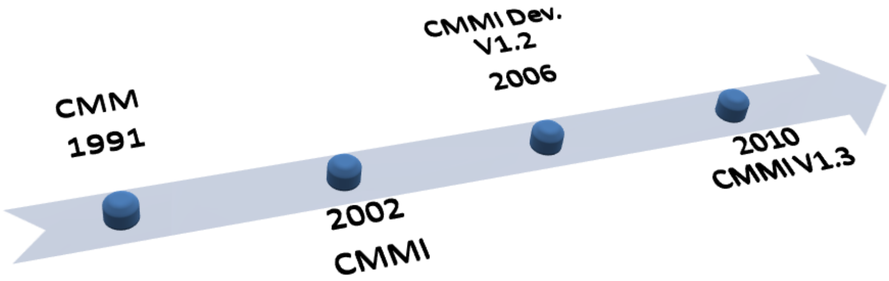


Figure 3.1: Subset of the CMM(I) releases.

3.1.2 Nature of CMMI and TSP

Herbsleb and Goldenson (1996) conducted a survey about the experience and results of CMM certified organisations, showing a relationship between high maturity level and being able to meet the schedule, budget and having higher staff morale. They also showed a tendency of having better quality, productivity and customer satisfaction, with a "consistency highly unlikely by chance alone". The implementation of the CMMI maturity level 5 includes a set of demonstrated benefits for the organisations (Goldenson et al., 2004):

- Improve the quality of the products and processes;
- Reduce the development cost and the schedule;
- Increase the customer satisfaction;
- Add predictability to the processes;

[1] Deming espouses the Shewhart concepts (Humphrey, 1992).

[2] Plan, Do, Check, Act cycle, is the foundation of process improvement work (Humphrey, 1992).

- Reduce rework by reducing the quantity of defects detected later in the life-cycle and which imply spending time finding and correcting them.

However, not all organisations achieve the same results. The 2003 TSP report indicates that the defect density (number of defects per a thousand lines of code) in products delivered is lower in organisations using TSP than in CMM level 5 organisations. The evidence is graphically represented in Figure 3.2, which shows the number of defects per a thousand lines of code (KLOC) delivered to the customer.

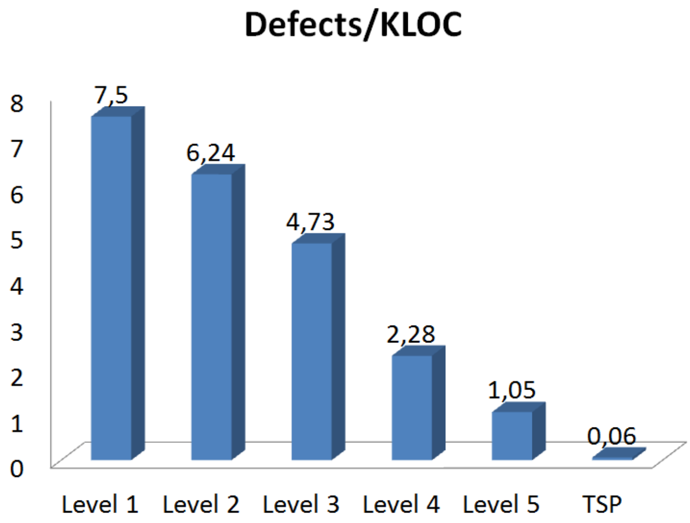


Figure 3.2: Average defects per thousand lines of code of delivered software in TSP and CMM different maturity levels (Davis and McHale, 2003).

From these results we could consider that since TSP performs better in quality than CMM, it would be preferable to use TSP rather than CMMI. However, the question is whether they are even comparable. Surveying CMM organisations, Herbsleb and Goldenson (1996), indicated they were able to identify and understand what needed to be improved but considered they did not have enough "guidance on how to improve". In fact, regarding CMMI, the model is not prescriptive and for that reason the organisations using it present different performance results - such variance depends on the methods used to implement it. Furthermore, TSP only covers part of the CMMI practices.

The Accelerated Improvement Method (AIM) implementation guidance (McHale et al., 2010) provides information on TSP evidence that allows achieving CMMI ML3, but ML4 and 5 are not yet covered. The AIM is a process improvement initiative that combines the best of CMMI, TSP and Six Sigma measurement and analysis techniques. TSP implements 70% of the specific practices up to maturity level 3 (CMU/SEI, 2010c).

Webb et al. (2007) extended TSP to address directly the CMMI process areas that are not addressed completely by TSP practices by publishing additional process scripts, items that were added to existent processes, metrics and requirements.

PSP and TSP are the application of high maturity to teams (Davis and McHale, 2003), while CMMI defines the capability of an entire organisation. They have different natures and purposes: CMMI gives guidance and organisations are free to select the most adequate methods to implement the practices; TSP provides all necessary tools to have a mature process in place and improve it. To help us understand why organisations using CMMI can have different performance, and sometimes even an unacceptable one, we analyse in the next section the problems that can arise while implementing process improvements, metrics programmes and CMMI.

3.1.3 Problems in Process Improvements, Metrics Programs and CMMI

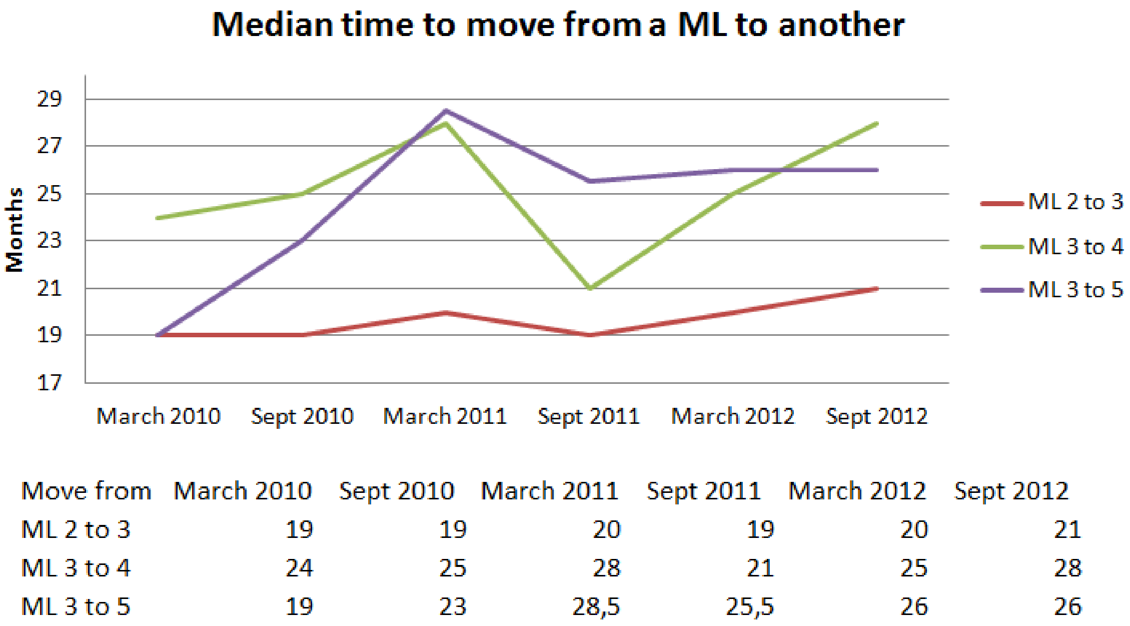
The results of a 1996 survey (Herbsleb and Goldenson, 1996), about the experience and results of CMM organisations, indicated that 26% of the organisations stated that since the implementation of the model nothing much changed, and 49% said they were disillusioned with the lack of results. Later on, a survey conducted to understand what CMMI level 4 and 5 companies used in the CMMI implementation showed that practices were not clearly institutionalised (Radice, 2000). The SEI concluded that some companies did not understand the statistical nature of the CMMI level 4 and certified CMMI HML companies did not have a consensus on the necessary characteristics of level 4 (Hollenbach and Smith, 2002). Charette et al. (2004) reported issues such as lack of capability, poor performance, and/or lack of adherence to processes that were found in the application of CMMI. The performance of high maturity organisations was questionable (Bollinger and McGowan, 2009).

In 2004 the Department of Defence (DoD) of the United States of America (USA), pointed out problems that result from having CMMI levels (Schaeffer, 2004). Among other problems, it was recognised that not all programmes of the organisations are appraised. Therefore, practices were not implemented organisation wide and organisations let the baselines erode once they achieved a certain maturity level. In response to the DoD problem, Pyster (2006) proposed a set solutions, of which the following are examples (the ones related to acquisition are identified with the word in parentheses):

- Guaranteeing that when contracting an organisation with a certain maturity level, the performing team uses the maturity processes being referenced (acquisition);
- Doing periodical appraisals after the contract to ensure that the tailoring of adequate processes for the specific programme include adequate high maturity processes (acquisition);
- Recognising that CMMI is relatively new and will take time to "fully permeate companies";
- Improving the appraisals by providing guidance on how to select representative samples and aggregate results from subordinate organisations, when appraising large organisations.

The last suggestion that Pyster made may improve the SCAMPI and avoid the certification of organisations where the practices are not institutionalised. The other suggestions are important for the contractors that demand that their suppliers have a specific CMMI maturity level, but do not tackle the root problem that may exist either in CMMI or in SCAMPI, or perhaps in both.

CMMI implementation takes time and organisations may not be ready to dedicate their most valuable people for such a long period of time to work in SPI or consider that they are able reduce that time. The results of Herbsleb and Goldenson (1996) showed that in 42% of the organisations the programme was overcome by higher priority events or critical issues. Furthermore, 77% of the surveyed organisations stated that it took longer than initially planned and nearly as expected, and 68% indicated they exceeded the estimated costs. The reported median times to move from level to level are presented in Figure 3.3 (CMU/SEI, 2010a,b, 2011a,b, 2012a,b).



Move from	March 2010	Sept 2010	March 2011	Sept 2011	March 2012	Sept 2012
ML 2 to 3	19	19	20	19	20	21
ML 3 to 4	24	25	28	21	25	28
ML 3 to 5	19	23	28,5	25,5	26	26

Figure 3.3: Median time to move from a ML to another based on semi-annual SEI reports from 2010 to 2012 (CMU/SEI, 2010a,b, 2011a,b, 2012a,b).

Some organisations opt to try to be certified at ML 5 straight from ML 3, instead of doing progressive certification. It is interesting to see that in some semesters the time to move from ML 3 to 4 exceeds the needed time to move from ML 3 to ML 5. These results depend on the maturity and readiness of the organisations, which are doing the implementation and being appraised in that period.

Many companies face problems when implementing CMMI HML that arise from complex practices such as measurement and quantitative management or the use of effective performance models for predicting the future course of controlled processes. In fact, part of the difficulties found in the processes evolution and new Process Areas implementation are related to the need to move towards a statistical thinking and quantitative management (Takara et al., 2007).

Kitchenham et al. (2006) analysed a CMMI level 5 corporation's database, and found that data were collected but metrics could not be correlated and did not have meaning for upper management. According with Monteiro et al. (2010), some authors (Hall et al. 1997; Berander and Jönsson 2006; Agresti 2006) argue that several measurement programmes in organisations fail because they define too many measures that are not actually implemented and analysed in decision making. Kitchenham et al. (2006) disclosed important concerns that shall be taken into consideration when storing data and what needs to be considered when designing the database, so data analysers and decision makers can actually make use of them. They also proposed the use of the M3P framework that extends the GQM (Goal Question Metric) by providing links between the collected metrics, the development environment and the business context.

Regarding the metrics definition, it is important to understand how the data is collected and analysed, what are the common and special causes of variation. As already mentioned many metrics exist which are either expressed in natural language, or the values that allow their calculation are expressed in natural language (Goulão, 2008). Breuker et al. (2009) mention there are different definitions of the same software metrics in the literature (books and papers), tools for metrics collection's specification and tools that actually collect the metrics. Literature needs to clearly define software metrics and practitioners should be aware of this problem when implementing the measurement and analysis system.

Leeson (2009) added more problems that can occur in CMMI implementation, which we have compiled in table 3.1. Those problems are classified as Program Management to refer to the management of the implementation of CMMI, Maturity Level 2 and Maturity Level 3, to refer to problems that occur in the implementation of those ML.

In her doctoral thesis, Barcellos (2009) states that metrics programmes are failing because they are producing metrics that do not allow the analysis of the performance and capabilities of their processes (Goh et al., 1998; Fenton and Neil, 1999; Niessink and Vliet, 2001; Gopal et al., 2002; Wang and Li, 2005; Kitchenham et al., 2006; Sargut and Demirörs, 2006; Curtis et al., 2008; Rackzinski and Curtis, 2008). In her literature review, she also states that there are problems of metrics adequacy (Wheeler and Poling, 1998; Kitchenham et al., 2006; Tarhan and Demirors, 2006; Boria, 2007; Kitchenham, 2007; Tarhan and Demirors, 2006; Boria, 2007; Kitchenham, 2007; Curtis et al., 2008; Gou et al., 2009).

The SEI and the Systems and Software Consortium, Inc. (SSCI) published a report in 2009 with the reasons contributing to the success of CMMI programmes (SEI and SSCI, 2009). The authors concluded that people acting as individuals or as teams contribute to programme success or failure. The main points were decision making, communication, teams experience, adequate coaching and understanding the programme purpose and goals. It is recognised that a good process is not enough for a programme to succeed and the factors that are considered as overriding the above mentioned are "effective leadership and objective governance for the programme" and "willingness and ability of programme personnel to think through problems and tailor the prescribed process to the needs of the programme".

Table 3.1: Some of the problems identified in the implementation of CMMI (Leeson, 2009).

Problem Type	**Description**
Program Management	Senior management is not involved in establishing the objectives, policies and the need for processes.
	Sponsor does not play its role and delegates authority.
	Software Engineering Performance Group is not managed.
	Organisations are focused on achieving a maturity level more than improving the quality of their products or services.
Maturity Level 2	Organisations lack a global view of the model. Organisations do not understand the relationship and differences between measurement and project monitoring, GP (Generic Practice) 2.8, 2.9 and 2.10, process areas and practices, maturity levels and capability levels.
	Some organisations misinterpret ML 2 and 3, which causes the failure of many programmes.
	Assume that ML2 is only for project managers. Developers and engineers need to be involved.
	Measurements are not related to customer and business objectives.
	Quality Assurance is focused on product compliance instead of assuring the quality of the processes.
Maturity Level 3	Some organisations define theoretical processes that do not correspond to actual activities to achieve ML 3.
	A communication process is not established, such that experiences and suggestions from people who know how to do their job are not gathered and consequently are not fed back.
	Organisations do not consider HML, when implementing ML 3, and fail to understand the end-picture, not seeing the direction they are taking at lower levels before moving to HML.

At this point we have partially answered research question **RQ1** - *Why do some organisations not achieve the expected benefits when implementing CMMI?* Even though the SEI, and now the CMMI Institute, publish good performance results from organisations that implement CMMI process improvement programmes to ensure that CMMI users are benefiting from using the model, all the problems mentioned in this section are the result of a deficient implementation of CMMI. Such problems become more damaging when they are not detected in the appraisal.

3.1.4 SCAMPI Limitations

The SCAMPI appraisal method already missed some implementation problems. After analysing the description of the method, some of its features can be regarded as limitations that may be in the origin of this problem.

Appraisal Team Quality

Armstrong et al. (2002), in their presentation of the changes and features of the SCAMPI V1.1, stated that the appraisal method is focused on practices implementation. That is, the SCAMPI appraisal team is focused on verifying whether the practices are implemented or not. The objective of the appraisal is not to verify how people are actually doing the work or the quality of their results. With such an orientation, malpractices may be missed by the appraisal team. In fact, the appraisal results reflect the knowledge, experience and the skill of the appraisal team (CMU/SEI, 2011c).

Organisation Honesty

SCAMPI relies on the organisation's honesty, that provides evidence and supports the choice of the projects that are going to be appraised (Armstrong et al., 2002). Either the lead appraiser is very rigorous in the choice of the projects and critique about the evidence or the outcome of the appraisal may be biased by the organisation.

Limited Number of Affirmations

In the appraisal only a small number of affirmations sustain practices. In version 1.2 to classify a practice as fully implemented, and therefore contributing to the level achievement, a direct and indirect artefact could suffice (CMU/SEI, 2011c). Back in 2003, Radice described a 50% : 50% rule that stated that there should be an affirmation in 50% of the practices and 50% of the projects covering one row per one column. The SCAMPI V1.3 coverage rules, which we previously described, also limit the number of affirmations. We consider that an affirmation from a single Business Unit (BU) that does not come out in an interview because the coverage rules do not demand it, could suffice to demonstrate that the BU was not following one of the practices.

Coverage of the Organisation

As mentioned by the DoD, not all programmes of the organisation are analysed in the appraisal (Schaeffer, 2004). During the appraisal only a small percentage of the organisation projects and business units may be appraised, therefore it is difficult to have guarantees that the entire organisation is working in the same way in all projects or programmes – this in turn means that the practices may not be institutionalised. In face of the limitations of SCAMPI V1.1 description in providing guidance to the selection of the projects for the appraisal, Moore and Hayes (2005) proposed the application of Design of Experiments (DOE) to construct a representative sample of the organisational units being appraised. DOE is a statistical technique that helps understanding the influence of the different experimental factors on the response of the system. When applied to the SCAMPI the method allows an accurate appraisal planning and execution, as it supports the construction of a representative sample of the organisational unit and the selection of the personnel to interview and questions to be asked when collecting affirmations. The projects are selected from analysis of the influential factors of the organisation unit. In order to reduce the number of projects the authors propose the use of fractional factorial design of instantiations and to apply the method in sequence (early SCAMPI C's and later SCAMPI A's or B's) in order to eliminate factors based on results. Later, Moore and Hayes (2006) presented useful information on how to use the previously mentioned method, DOE, to select the most appropriate projects for an appraisal, fulfilling the SCAMPI V1.2 description. SCAMPI V1.3 clearly defines the sampling rules.

Evidence Collection

Pricope and Horst (2009) indicated that the SCAMPI is described in natural language and does not provide activity-oriented graphical description of the appraisal process. For those reasons the authors proposed a method to measure SCAMPI appraisals by using the Unified Modelling Language (UML) to represent the metamodel of the SCAMPI. The metamodel includes all the SCAMPI elements, such as types of evidence, activities performed in the appraisal, roles, etc. Besides the model, the authors proposed quality metrics to evaluate the appraisal, introducing a quality metric for activities. The metrics allow determining the level of weakness or strength of the appraisal elements. The method proposed by Pricope and Horst introduced the quantitative novelty in the SCAMPI and is useful for quantifying the appraisal that was conducted. However, it does not evaluate how practices are actually being done nor evaluates the organisation performance.

Sunetnanta et al. (2009) proposed a model that constitutes a Configuration Items (CI) repository, where all projects configuration items are pooled together. The authors' idea was to use this repository in organisations working with different offshore units, however we consider that the model is applicable to any organisation. The CMMI appraiser needs to set up the rules to identify the projects' CI that constitute evidence of the sub practices of the model. The repository allows the collection of evidence as the projects are ongoing, as well as analysing the projects and appraisal results. The quantitative assessment of the projects is done by score, i.e. number of times an activity is executed, and by compliance, i.e. by checking if the activity was executed or not. By the time of the appraisal all the evidence is already available. There is a limitation on the method that the authors do not mention, though. The evidence still needs to be evaluated and analysed by the appraisal team. A CI may be generated but if it is empty it shows that the expected activity was not performed, and even when generated and is not empty, it is still necessary to assess whether people actually carried out the practice or just produced an artefact that is mandatory.

Pilots on Results-based Appraisals

McCarthy (2009) reported results of pilot projects being assessed with results-based appraisals, including the Telecommunication Quality Management System – TL 9000 standard, to identify and validate an appraisal method that would assess performance measures. The information collected on SCAMPI would be useful to trigger the appraisal team for further investigation in face of unexpected performance, have results-oriented findings, have records for posterior assessments and recommendations related to performance and benchmarking. From the identified challenges the appraisals took longer (more 5% to 10% when compared with a regular appraisal) and became more expensive. The industry benchmarks varied in value which raised doubts as to their applicability. Besides, the measurement repository built in the pilot environment had no documented linkage to processes and practices in standard process or in the CMMI.

CMMI has been facing the paradigm of how well high maturity organisations are performing for quite some time (Bollinger and McGowan, 2009; Campo, 2012). SEI released version 1.3 that included improvements in the model, focusing on the performance of the organisations (Phillips, 2010b), but SCAMPI still does not measure their performance.

3.1.5 CMMI V1.3 Changes

The CMMI product team worked on the definition of the version 1.3 of the CMMI constellations and the SCAMPI. The new version of the CMMI model was designed to be more compatible with the multi-model tendency that has been occurring. Figure 3.4 depicts the relationships between different models. To implement good practices, organisations follow quality principles, such as CMMI constellations, ISO standards and the Project Management Institute (PMI) documentation. To know how to follow the quality principles, organisations use operational practices, for example TSP, Agile and the Information Technology Infrastructure Library (ITIL). We consider that methods also include techniques and procedures used to generate the products and/or services. Organisations also use improvement techniques that help them evolve and shape their processes, such as Lean, Six Sigma and Theory of Constraints. We consider that those improvement techniques can also be used to define organisations processes.

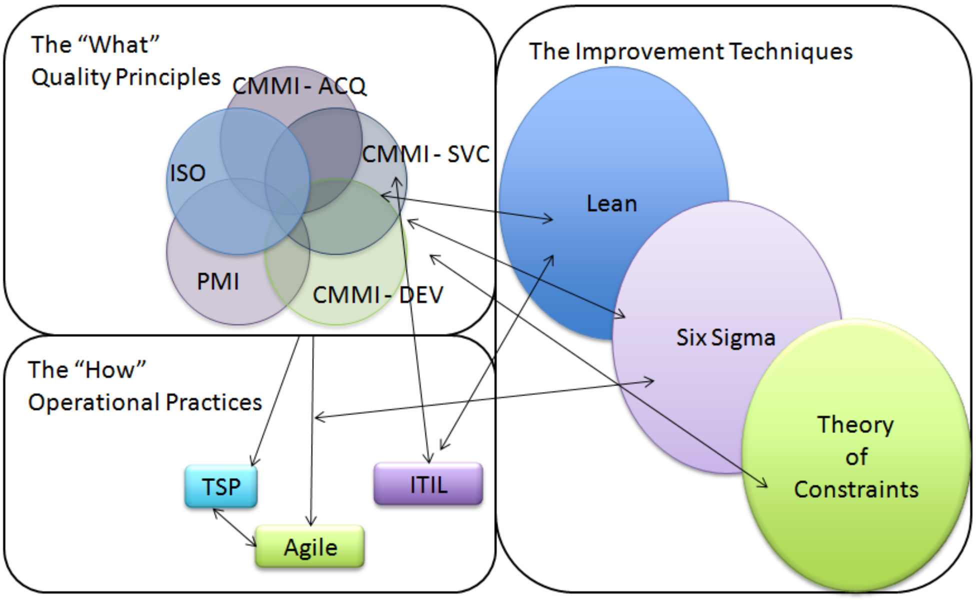


Figure 3.4: Multi-model representation (Phillips, 2010b).

With the new version of the CMMI, the SEI intended to make the following improvements (Phillips, 2010b):

- Simplify/clarify terminology;

- Update selected process areas to provide interpretation of practices for organisations with respect to Agile methods, quality attributes, product lines, systems of systems, customer satisfaction, amongst others;

- Simplify Generic Goals and remove the ones of the high maturity levels;

- Clarify High Maturity concepts, such as process models and modelling, business objectives thread to high maturity, common causes, high maturity expectations in individual process areas, amongst other changes;

- Add a new process area in ML 5, called OPM (Organisational Performance Management), that substitutes OID (Organisational Innovation and Deployment);

- Revise QPM (Quantitative Project Management) SP to reflect the connection between CAR (Causal Analysis and Resolution) and QPM;

- Lighten the model in terms of number of pages, generic goals and practices, and specific goals and practices.

Figure 3.5 represents the combination of OPM and OID that gave origin to OPM. With OPM the improvements are driven by the intention to achieve quantitative objectives of quality and process performance. The drivers of the improvements will not only be the organisation but also the customer.

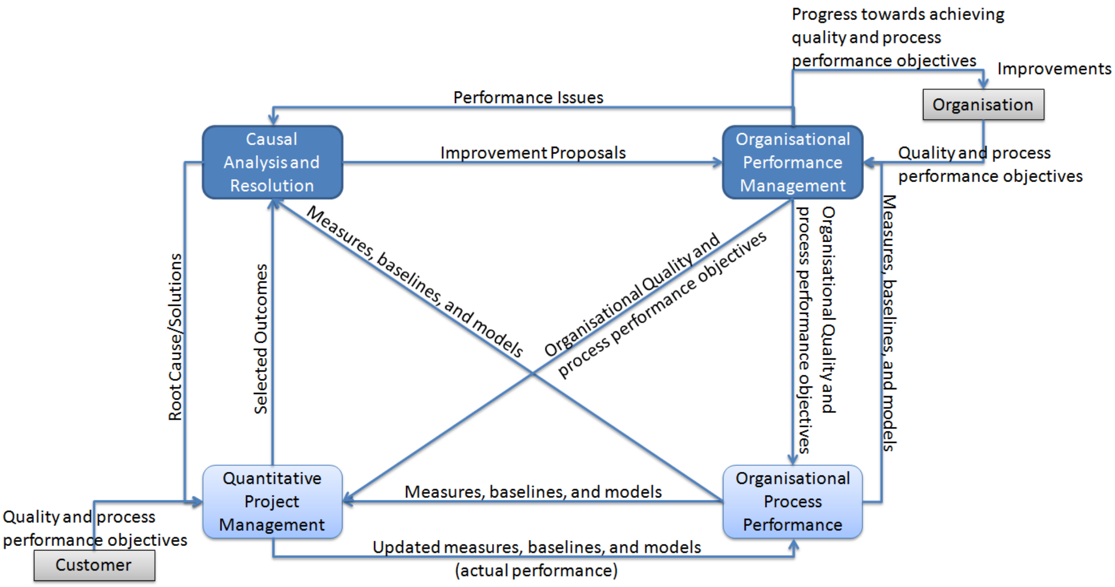

Progress towards achieving quality and process performance objectives
Improvements
Organisation
Performance Issues
Causal Analysis and Resolution
Improvement Proposals
Organisational Performance Management
Quality and process performance objectives
Measures, baselines, and models
Organisational Quality and process performance objectives
Organisational Quality and process performance objectives
Measures, baselines, and models
Root Cause/Solutions
Selected Outcomes
Quantitative Project Management
Measures, baselines, and models
Organisational Process Performance
Quality and process performance objectives
Customer
Updated measures, baselines, and models (actual performance)

Figure 3.5: Representation of OPM and OID (Phillips, 2010b).

Version 1.3 of the SCAMPI has the following changes (Phillips, 2010b):

- Includes details on the definition of the appraisal scope and on how to sample the organisation units and projects;

- Clarifies doubts (for example, about direct and indirect artefacts);
- Improves appraisal efficiency.

Regardless of the improvements in defining the scope and collecting enough evidence SCAMPI appraisers raised some limitations on the coverage rules. One example was given by Heather Oppenheimer, on August 2011[3], stating that for some support functions some activities of a PA are assumed by one group and others are used by basic units because they support their work. This means that the support function cannot provide all the evidences for the PA. We consider that there is also another problem with the coverage rules. It is possible to leave lack of institutionalisation undetected, because not all basic units need to provide artefacts/affirmations. Some of them may not be doing a PA that concerns their work at all.

The CMMI model evolved to focus on the improvements in the performance of the organisations, paying more attention to results. This may help to avoid implementations which are more oriented to achieving a maturity level rather than improving organisation's performance. The model was also updated to cope with multi-model environments. Having guidance for different models may prevent poor practice's implementation.

SCAMPI is addressing part of its problems, in particular by better defining the scope of the appraisal, i.e. how to sample from the organisational units and how much evidences are necessary. However, we consider that the implementation problems may remain undetected and the performance problems may thus persist, because the SCAMPI is not evaluating the implementation performance, which is out of its scope. For this reason it is necessary to do further research in this area in order to have a framework that measures organisations performance and evaluates the quality of implementation of the CMMI practices. With the analysis we presented in the current and previous subsections, we complete the answer to research question **RQ2** - *Why does SCAMPI not detect implementation problems, or does not address performance evaluation in all maturity levels?*

3.1.6 Methods and Models for Process Measurement and Evaluation

Next we present other research contributions that may help in understanding and solving the problem addressed in this work. With this analysis we clearly define what is still needed to tackle the problem and motivate our approach.

Metrics Definition

In 2009 the SEI and the Object Management Group (OMG) announced the creation of the Consortium of IT Software Quality (CISQ) (CMU/SEI). The CISQ is sponsored by OMG and the SEI is working with them in the development of software-related standards and appraiser licensing

[3] http://www.linkedin.com/groupAnswers?viewQuestionAndAnswers=&discussionID=64637798&gid=54046&commentID=54099629&trk=view_disc&ut=0b8g1zukIS5R01 – last accessed on 17-11-2011.

programmes. The metrics would be unambiguously defined, contributing to the possibility of automating the measurement and analysis process (Curtis, 2010). The consortium published code quality standards to be consistently applicable to any organisation: the Automated Function Points (AFP) (CISQ, 2014), as a standard measure of size, with rules to measure different software code files; and Automated Quality Characteristic Measures (CISQ, 2016), compliant with ISO/IEC 25010 quality characteristics of security, reliability, performance efficiency and maintainability, and providing "measures of internal quality at the source code level". CISQ is currently developing Automated Enhancement Points, a measure of size to be used in productivity analysis; Technical Debt, measuring the cost, effort and risk of the remaining defects in code at release; and Quality-Adjusted Productivity to consider the quality in the measurement of productivity.

An anonymous research group from Switzerland worked on the definition of metrics to assess the quality of the CMMI implementation. The results of their work were not published since they concluded that such task would be impossible to execute. Their justification for that conclusion is that it is difficult to define metrics applicable to all organisations, because each organisation has its own business objectives. A member of the group shared with us the unpublished documentation of their work (Anonymous Research Group, 2007). They had undertaken an exhaustive work identifying metrics that would allow the control of 11 process areas, which the researchers referred to as processes. One of the problems encountered in the research work done was that, instead of first reviewing the literature in search of the metrics for the processes that they intended to monitor, they opted to introduce new ones. There are very many metrics identified in software engineering, and most of them are never used. We noticed that the metrics are described but the goal that would support each metric had been ignored. In our opinion, if the metrics defined by the research group are of use they need to be mapped with the questions that the organisation would want to have answered in order to verify that a certain objective would be achieved. The way the metrics document was built reveals that the usage of the BSC and GQM were not considered.

With the existence of a size measurement standard, already released by CISQ, and future releases of productivity standards that take the quality of the developed code into account, organisations can collect data, the same way, making it comparable. Moreover, the fact that productivity will be based on the quality of the produced work makes the metric more useful and relevant. This will facilitate the task of benchmarking organisations performance, which in turn will help overcoming some of the aforementioned limitations of process performance assessments.

Metrics Analysis

The SEI Software Engineering Measurement and Analysis (SEMA) group (CMU/SEI, 2001; SEI, 2016) publishes the results of the state of measurement and analysis practices and conducts research of the most effective ways to implement measurement and analysis processes from the Process Area MA (Measurement and Analysis) at ML 2 to the high maturity techniques that are necessary to implement levels 4 and 5. SEMA developed the Quantified Uncertainty in Early Lifecycle Cost Estimation (Ferguson et al., 2011) that considers "programme change driver uncertainties common to programme execution in a DoD Major Defence Acquisition Program lifecycle". The method includes the use of Bayesian Belief Networks to generate likely scenarios and Monte Carlo Simulation to estimate the distribution of the programme cost.

The Performance Benchmarking Consortium (PBC) was created in 2006 with the objective of providing tools and credible data for goal-setting and performance improvement and combining data from different provenances to create a superset of information for benchmarking and performance comparison (Zubrow et al., 2006; Kasunic, 2006). The benefits of the initiative would be to establish the specifications for the collection and comparison of data from different source vendors and provide existing data to organisations to help them establishing and achieving their business goals. PBC members would contribute with their assets to a repository, and PBC would specify the measurements, in order to make the members' data comparable. The subscriber organisations would be able to access the repository, have access to performance reports and submit performance data adherent to the measurements specifications. A large database was to be built with performance data of several organisations, but the analysis of the metrics allowed to conclude that they were not comparable because the organisations had their own definitions of each measure. They tried to use the TL9000 standard to define metrics but the standard has too many, and this would not be acceptable in the software industry (Phillips, 2010a, personal communication). The useful result of the consortium was the publication of the data specification for software process performance measures (Kasunic, 2008). From the experience of running pilot projects of SCAMPI appraisals oriented to results (see 3.1.4 SCAMPI Limitations) and an attempt to create a projects database with the PBC, the necessity of having a standard of software engineering metrics is clear.

The Software Productivity Research (SPR) is a consulting services provider, created by Capers Jones. Capers Jones analyses data related to software processes performance, gathered by several organisations in USA. In his work he classifies software development projects in categories and supports the choice of the adequate quality decisions according with the characteristics of the projects, based on data (Jones, 2010).

The Software Benchmarking Organisation (SBO) used data gathered by Capers Jones (2008) to apply benchmarking in the comparison of the behaviour of European projects with USA data. The results showed that projects of European organisations behave similarly to the USA's projects. Sassenburg and Voinea (2010) identified Key Performance Indicators (KPI) that would:

- Support project management in analysing, planning and monitoring projects;
- Inform top management of the status of the project and the direction that it is heading;
- Support business units in measuring their capability improvements;
- Support organisations in comparing/benchmarking business units.

They have a set of questions that allow to identify KPI of different categories: project performance, process efficiency, product scope and product quality. We summarise them in table 3.2.

Table 3.2: KPI categories - based on (Sassenburg and Voinea, 2010)

Question	KPI Category	KPI
How predictable is the project?	Project Performance	Cost, schedule, staffing rate, productivity.
How fast is my process?	Project Efficiency	Effort distribution (cost of the quality model).
How much of the product?	Product Scope	Features, deferral rate, size, re-use.
How well are we doing?	Product Quality	Complexity, test coverage, removal efficiency, defect density.

SBO's results indicate that it is possible to apply benchmarking techniques to the processes performance data, and organisations data becomes comparable by these means. The benchmarking structure that is applied in SBO is done in three phases, each one of them involving a certain number of processes that are characterised by process definition elements (Sassenburg, 2009). The description of the method to perform benchmarking in organisations does not correspond to expectations when asking for the benchmarking process. One of the outcomes of benchmarking is the definition of a process that is applicable to any objects that we intend to compare. In this particular case, the objects would be organisations.

Evaluate Factors of Success

From a literature review, Jeffery and Berry (1993) extended a framework to evaluate and compare reasons for the success and failure of metrics programmes. They analysed several authors' recommendations for the success of metrics programmes and used Fenton's categories, namely context, inputs, process and products, to classify them. Based on that, they built a questionnaire to conduct a case study to analyse organisations in those perspectives. They provided the organisation's context and goals for the metrics programme. The framework also includes a scoring scheme, to classify the extent to which the criteria were met. Later on, Wilson et al. (2001) adapted the Jeffery and Mike framework to be used in SPI.

Niazi et al. (2005) created a maturity model to assess and improve the implementation process of SPI. They related critical factors with maturity stage based on their occurrence in the literature and considering the inputs from interviews that they conducted. Those factors are related to the way SPI is conducted and not to improving processes outputs.

Metrics Repositories

Palza et al. (2003) designed an object-oriented model named Multidimensional Measurement Repository (MMR) to collect, store, analyse and report measurement data in order to facilitate the implementation of CMMI. MMR is based on PSM and the Software Measurement Process (ISO/IEC 15939:2007).

The Alarcos research group proposed a measurement model (García et al., 2006) and an ontology for software measurement (Canfora et al., 2006). In 2007 they presented a proposal to support consistent and integrated measurement of software by providing the following elements:

the Generic measurement metamodel, to represent the data related to the measurement process and the GenMETRIC, a tool that allows the specification of software measurement (García et al., 2007).

The Quantitative Approaches on Software Engineering and Reengineering (QUASAR) research group worked on the unambiguous definition of metrics. They also applied metamodel based approaches, in particular by using Ontology Driven Measurement (ODM), as defined by Goulão (2008). This model was an evolution of the MetaModel Driven Measurement (M2DM) for the evaluation of object-oriented designs of software engineering metrics (Abreu, 2001). In 2009 the ODM method was being applied to SQL databases in a Master Science thesis supervised by Goulão. Metrics metamodels can be used in the unambiguous definition of our framework's performance indicators.

Software Process Measurement

The Alarcos research group applied a metamodel approach to the management of software process performance measurement. They developed an integrated framework to model and measure the software processes based on number of activities, steps, dependencies between activities of the process, activity coupling, number of work products, number of process roles, and so on (García et al., 2003). In our research, process measurements are not restricted to the process itself but include the performance (efficiency and effectiveness) of its outcomes.

The Integrated Software Acquisition Metrics (ISAM) is an SEI project that consists of developing a common measurement framework for acquirers and developers based on TSP and PSP practices. Our framework not only measures CMMI practices, but also evaluates the quality of their implementation.

Process Modelling or Simulation

Hsueh et al. (2008) showed that UML and software simulation can be used in the design, verification and validation of processes. They designed a static process metamodel that establishes the relations between the elements of CMMI and process components. Figure 3.6 presents the metamodel of CMMI. In their work, static processes are modelled with class diagrams and include the relationships between process elements and processes in CMMI. Process elements behaviour is modelled using state-chart diagrams. Finally, dynamic processes sequences are modelled with activity diagrams. The process verification is done by the definition of process rules and CMMI verification rules, using the Object Constraint Language.

Mishra and Schlingloff (2008) proposed a formal specification based product development model that integrates product and process quality to the implementation of processes. They demonstrated the model in process compliance with CMMI, without considering performance.

Process Modelling and Measurement

Colombo et al. (2008) designed a metamodel to support multi-project process measurement to calculate across-process-multi-projects metrics aligned with CMMI. They developed an open source tool named Spago4Q that supports different development models, such as waterfall and

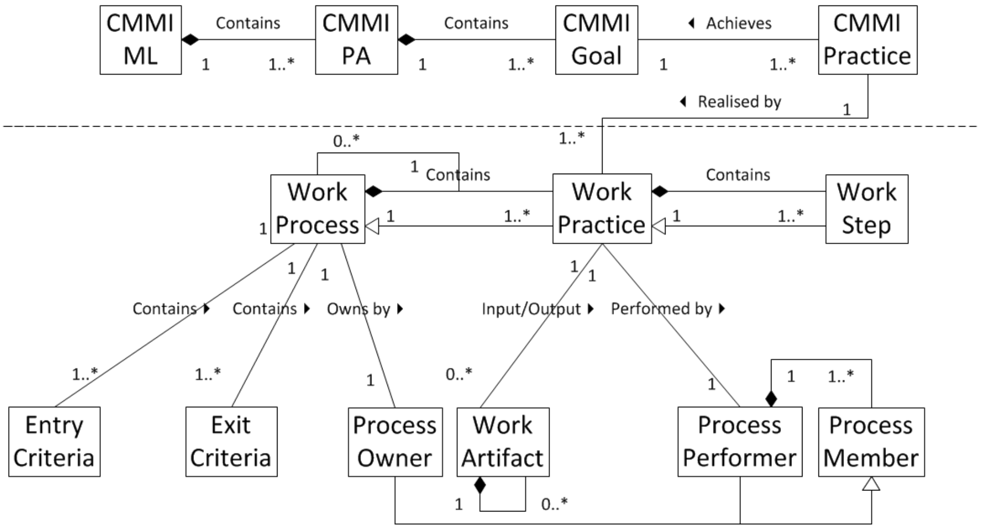


Figure 3.6: CMMI static process metamodel (Hsueh et al., 2008)

agile. The CMMI assessment framework is supported by the Assessment component of Spago4Q. The tool is available online[4] and is announced as being "a platform to measure, analyse and monitor quality of processes, products and services". Such a tool may be useful to support the implementation of our framework.

In table 3.3 we summarise the existent frameworks and give some comments about them. The authors are identified by their surname or research group.

Table 3.3: Related Frameworks.

Framework Type	**Description**	**Comments**
Metrics Definition	Standard of code size metric that allows automation (CISQ).	Useful to have common and unambiguous metrics definition. A productivity metric that considers quality, yet to be released.
	Metrics for CMMI Process Areas (Anonymous Research Group).	Too many metrics unrelated to goals.
Metrics Analysis	Performance Benchmark Consortium (Kasunic).	Metrics not comparable due to different definitions.
	Publication of projects data (Jones).	Too many metrics unrelated to goals.

Continued on next page

[4] http://www.spagoworld.org/xwiki/bin/view/Spago4Q/ - last accessed on 29-05-2011.

Table 3.3 – *Continued from previous page*

Framework Type	Description	Comments
	Using projects data for benchmarking KPI (Sassenburg and Voinea).	KPI that can be used to evaluate practices at higher level and characterise organisations performance.
Evaluate Factors of Success	Based on questionnaire and score. Used to evaluate: - Metrics Programs (Jeffery and Berry; Wilson et al.); - Software Process Improvements (Niazi et al.).	Based on how programmes (SPI, Metrics) are conducted.
Metrics Repository	Data model of software development (Kitchenham et al.).	Model for a metrics database considering context.
	Multidimensional Measurement Repository (Palza et al.; García et al.). Metamodel design for software engineering metrics (Canfora et al.; García et al.). Ontology for measurement and a measurement tool (Goulão).	Formal specification of metrics and building repositories.
Software Process Measurement	Model and measure software processes (García et al.). Measurement framework for TSP and PSP practices (Nichols et al.).	Based on process structure characteristics.
Process Modelling or Simulation	Models/metamodels of CMMI based on UML (Hsueh et al.; Mishra and Schlingloff).	Focused on compliance.
Process Modelling and Measurement	Metamodel to support multi-project process measurement aligned with CMMI (Colombo et al.).	Support CMMI assessment, not particularly focused on the quality implementation of the practices.
SCAMPI Modelling	Graphical representation of the SCAMPI and quantitative evaluation (Pricope and Horst). Repository for distributed projects to collect evidence as they are being executed (Sunetnanta et al.).	Focused on compliance.
Performance Measurement	Performance measurement on SCAMPI for CMMI organisations benchmarking (McCarthy).	Introduced overhead on SCAMPI.

The frameworks mentioned so far can tackle some of the issues regarding the identified problem. However, they do not directly tackle the performance of process improvements. Also, one of the limitations verified in software engineering is there are several data repositories fed by different organisations, but not all are reliable, as organisations should provide the same metrics under the same definition, collected systematically; but they do not. The AFP standard allows organisations to automate their data collection on code size. If organisations use it and make their data available in public repositories it would be an excellent source for this area of research. In the mean time, organisations using TSP already collect data using a similar measurement protocol and metrics definitions, as TSP includes a set of forms to collect relevant metrics, used to evaluate the quality of the process. Therefore, EQualPI includes a 4.6.1 Data Dictionary with the definition of variables to collect in the scope of the demonstration of the Framework, a 4.6.2 Domain Model to describe the relation between the variables, and we validated EQualPI using TSP data (see details in section 5.3 Evaluation of the Estimation Process).

3.2 Survey on MA Performance in HML Organisations

Knowing that organisations can face several challenges when implementing CMMI, in particular when trying to achieve HML, what can they do to ensure that they properly achieve the desired high maturity goal? This question is indirectly answered in the reports of two surveys conducted by the SEI: TR2008 (Goldenson et al., 2008) and TR2010 (McCurley and Goldenson, 2010). The surveys, regarding the use and effects of measurement and analysis in HML organisations, were focused on the value added by PPM and the results were considered comparable. In 2008 the respondents were the sponsors (assisted by their delegates), or their delegates, from organisations appraised at CMMI HMLs. In 2009 similar questions were asked to lead appraisers of organisations pursuing HMLs.

The SEI analysis is relevant to organisations pursuing any ML of CMMI because the reports include problems and good practices that may have helped the organisations achieve HML. In one question, organisations had to indicate their routine while using PPM. The 2009 respondents indicated their most common problem was the long time it takes to accumulate historical data, which some organisations addressed by doing real time sampling of processes when they had no prior data available. In both surveys, respondents gave different importance to obstacles found in the implementation of PPM (Figure 3.7).

To measure the strength of the relationship between two variables and the accuracy of predicting the rank of the response, the authors used the Goodman and Kruskal's gamma. In 2008 the gamma value between the quality of the managers' training and their capability to understand PPM results was not very strong. However, that relation was stronger when the training was more formal. 80% of respondents considered that the builders and maintainers of PPM understood CMMI's definition of PPM and PPB very well or better, but their perception of the circumstances under which PPM and PPB are useful was lower. The results improved in 2009; over 50% of appraisers considered that the builders and maintainers of PPM understood all concepts very well or extremely well. We summarise the relations between dependent and independent variables and gama found on both surveys and results comments in Table 3.4. We also include Table 3.5 with the set of variables that was considered related with the achievement of HML.

Table 3.4: Summary of the surveys results TR2008 and TR2010.

Factors	**Dependent Variable**	**Independent Variable**	**Year**	**Gamma**	**Comments**
Training	PPM overall value	Managers training	2008	0.30 moderate	Discordant patterns in this relationship justify the low value of gamma.
	Stakeholders involvement in goal setting	Project Managers training	2008	0.31 moderate	In organisations with more formal training for managers the stakeholders are more involved in setting MA goals.
	PPM overall value		2009	0.66 very strong	The majority of respondents considered the training as good or excellent.
Tools	PPM overall value	Automated support for MA	2008	0.42 moderately strong	Automation is likely to payoff in better modelling outcomes.
			2009	0.45 moderately strong	Better than in the previous survey.
Healthy Ingredients (HI)	PPM overall value	Emphasis on healthy PPM ingredients	2008	0.55 very strong	Organisations whose PPM put emphasis on HI attributed more value to their modelling efforts.
			2009	0.66 very strong	Better than in the previous survey.
		Use of healthy PPM ingredients	2008	0.61 very strong	More organisations used PPM for purposes consistently with HI.
			2009	0.82 very strong	Better than in the previous survey.
Models and Analytical Methods	PPM overall value	Diversity of models	2008	0.57 very strong	Organisations using richer and varied suite of PPM were more likely to find value in their modelling.
			2009	0.57 very strong	Same as in the previous survey.
		Use of statistical methods	2008	0.54 very strong	Organisations could use more statistical methods, the ones using them find more value in PPM.
			2009	0.53 very strong	1/3 of the organisations use few statistical methods.
		Use of optimisation methods	2008	0.44 moderately strong	Few organisations used more than one optimisation method.
			2009	0.45 moderately strong	Essentially the same as in the previous survey.
		Data quality and integrity checks	2008	0.45 moderately strong	Respondents that do quality and integrity checks find their models more valuable.
			2009	0.49 moderately strong	A small improvement from the 2008 survey.

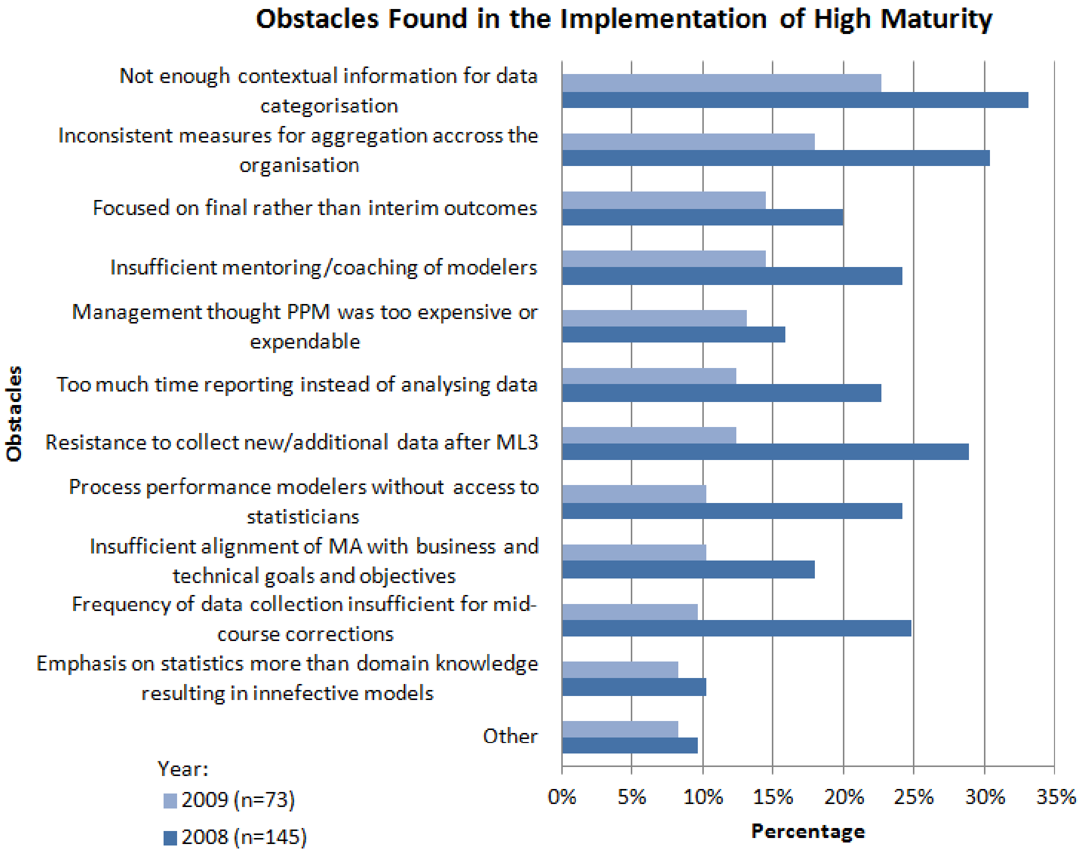


Figure 3.7: Obstacles identified by the organisations respondents found in the implementation of HML (TR2010).

Table 3.5: Variables related with achieving HML and TR2008 and TR2010 surveys results comments.

Variable	Mann-Whitney	Comments
Use of statistical methods	p < 0.001	Association between achieving target HML with moderate to extensive use of statistical methods
Documentation of PPM and measured results	p < 0.0000	Better documentation of process performance and quality measurement results in organisations that achieved HML
Emphasis on healthy PPM ingredients	p < 0.002	Closely related to achievement of HML
Use of healthy PPM ingredients	p < 0.0001	Comparable relationship with achievement of HML is extremely strong
Number of simulation/optimisation techniques used	p < 0.0000	Extremely strong relationship between the use of these analytical methods and achievement of HML
Use of PPM predictions in status and milestone reviews	Not given	Still quite strong relationship. Organisations that achieved HML used PPM in decision making more often than the ones who did not achived their target ML.

Organisations reported difficulties in collecting data manually. Regarding the automated support for MA activities, responses to the 2008 survey showed that the organisations used spreadsheets, automated data collection and management software and, less frequently, statistical packages, workflow automation or report preparation software. Automation, data quality and integrity checks, and the use of simulation and optimisation methods had a moderately strong relationship with the overall value of PPM.

The following list summarises the variables that had a very strong relationship with the PPM overall value first, followed by the ones that had a moderately strong relationship:

- Quality of the measurement training particularly for Project Managers;
- Models with emphasis on "healthy ingredients" (listed next), and models for purposes consistent with those ingredients;
- Diversity of models used to predict product quality and process performance;
- Use of statistical methods: regression analysis for prediction, analysis of variance, Statistical Process Control (SPC) charts, designs of experiments;
- Automation;
- Data quality and integrity checks;
- Use of simulation or optimisation methods: Monte Carlo simulation, discrete event simulation, Markov or Petri-net models, probabilistic models, neural networks, optimisation.

The SEI compiled a set of "healthy ingredients" to be considered in process performance modelling (Goldenson et al., 2008):

- Modelling uncertainty in the model's predictive factors;
- Ensuring models have controllable factors and possible uncontrollable factors;
- Identify factors directly associated with sub-processes to construct the models;
- Predicting final and interim project outcomes;
- Using confidence intervals of the expected outcome to enable "what if" analysis;
- Enable identifying and implementing mid-course corrections during projects execution towards successful completion.

When analysing the relations between the achievement of HML and certain practices some revealed differences between achieving and not achieving HML:

- All organisations with poor or fair documentation relative to process performance and quality measurement results failed to achieve high maturity, whilst most of the organisations with excellent and good documentation achieved HML;

- Using simulation/optimisation techniques had a strong relationship with HML achievement. The relation with the number of such methods used and achieving HML was very high. Particularly, all organisations using two of those methods achieved HML;

- There was a very strong relationship between achieving HML and, respectively: having models with emphasis on healthy ingredients, and the models for purposes consistent with those ingredients;

- All organisations that used statistical techniques substantially, achieved HML;

- The frequency of using PPM predictions in status and milestones reviews had a quite strong relationship with the achievement of the target HML.

In general the gamma between variables was higher in the 2009 survey. McCurley and Goldenson (2010) justify the improvements from the 2008 to the 2009 surveys results: "There may be a trend over time" and/or "The perspectives of the sponsors or the appraisers are more accurate". The results of both surveys indicate several improvements that HML organisations may consider to get full advantage of having PPM in place, but also show the obstacles that organisations that intend to implement HML practices may encounter.

3.3 Defect Classification Taxonomies

To define and validate the improvement component of the EQualPI framework (4.4.3 Process Improvements) we developed in our research, we conducted an experiment to improve the requirements review process, by introducing a classification specific for requirements defects (details in 3.3 Defect Classification Taxonomies). The literature reviewed that supported us on the definition of the classification scheme is discussed in this subsection.

In 2009, Chen and Huang performed an e-mail survey concerning several software projects, and presented the top 10 higher-severity problem factors affecting software maintainability, as summarised in Table 3.6. The authors indicated the following causes of software defects:

- a significant percentage of defects is caused by incorrect specifications and translation of requirements, or incomplete ones (Apfelbaum and Doyle, 1997; Monkevich, 1999);

- half of the problems rooted in requirements are due to ambiguous, poorly written, unclear and incorrect requirements, the other half result of omitted requirements (Mogyorodi).

In 2003, Lutz and Mikulski analysed the impact and causes of requirements defects discovered in the testing phase, resulting from undocumented changes or defects in the requirements, and proposed guidelines to distinguish and respond to each situation. Their work emphasises the importance of requirements management. Considering the problems that occur in the requirements specifications we next present work that is related with or includes a requirements defects classification.

Table 3.6: Top 10 Higher-severity problem factors impacting software maintainability (Chen and Huang, 2009).

#	Software Development Factors	Problem Dimension
1	Inadequacy of source code comments	Programming Quality
2	Documentation obscure/untrustworthy	Documentation Quality
3	Changes not adequately documented	Documentation Quality
4	Lack of traceability	Documentation Quality
5	Lack of adherence to standards	Programming Quality
6	Lack of integrity/consistency	Documentation Quality
7	Continually changing requirements	System Requirements
8	Frequent turnover within the project team	Personnel Resources
9	Improper usage of techniques	Programming Quality
10	Lack of consideration for software quality requirements	System Requirements

Code Defects Classifications, 1992

The Orthogonal Defect Classification (ODC) is applicable in all the development phases except the requirements phase. The defect types used are: function, interface, checking, assignment, timing/serialisation, build/package/merge, documentation and algorithm. For each defect it is necessary to indicate if the feature is incorrect or missing (Chillarege et al., 1992). Such classifiers do not seem completely adequate in order to classify requirements defects, and **Documentation** is too generic to give further information on the defect. Hewlett-Packard (HP) (Grady, 1992) categorises the defects by mode, type and origin, (see Figure 3.8). From the types of defects with their origin in the requirements specification phase, the requirements specifications seem to be vague and the interfaces ones are too detailed and more appropriate to design specification defects.

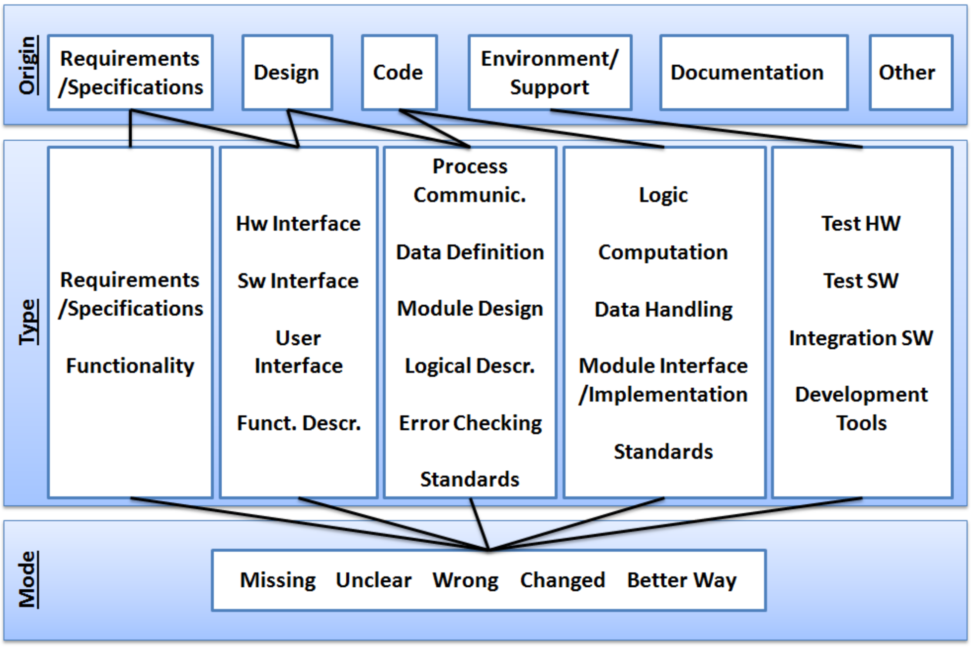


Figure 3.8: HP defects classification scheme (Freimut et al., 2005).

Quality Based Classifiers, 1976 – 2010

In 1976, Bell and Thayer conducted a research to verify the impact of defects in software requirements. Not surprisingly, they concluded that software systems meeting defective requirements will not effectively solve basic needs. They aggregated the defects in categories, as presented in Figure 3.9. In 1981, Basili and Weiss categorised defects found in requirements documents and gathered a set of questions to be asked while reviewing them (as a review checklist). Figure 3.9 shows the distribution of the 79 errors by different categories. Later, in 1989, Ackerman et al. analysed the effectiveness of software inspections as a verification process. They presented a sample requirements checklist to use in inspections of requirements documents, containing questions organised by defect categories: completeness, consistency and ambiguity. Then in 1991, Sakthivel performed a survey about requirement verification techniques and presented a requirements defects taxonomy based on a literature review (Walia and Carver, 2007). The classes that the author proposed are: incomplete, inconsistent, infeasible, untestable, redundant and incorrect. For each class, Sakthivel presented different defects and an example.

Hayes (2003), developed a requirements fault taxonomy for NASA's critical/catastrophic high-risk systems. Hayes stated that ODC refers to design and code while their approach emphasised requirements, so they adapted the Nuclear Regulatory Commission (NRC) requirement fault taxonomy from NUREG/CR-6316 (1995). Afterwards, in 2006, Hayes et al. analysed a software product related with the previous one to build a common cause tree. In both works *Unachievable* was reserved for future use. In 2006, the same was also done with *Infeasible* and *Non verifiable* (Figure 3.9 shows their results).

Defects classification is important to support the analysis of the root causes of defects. In 2010, Kalinowski et al. were aware that Defect Causal Analysis (DCA) could reduce defect rates by over 50%, reducing rework, and improving quality and performance. To enhance DCA, they improved their framework named Defect Prevention Based Process Improvement (DPPI) used to conduct, measure and control DCA. The authors mentioned the necessity of collecting metrics for DCA and the importance of considering:

1. Context when collecting metrics;
2. Stability of the inspection;
3. Technology/similarity of projects in inspections.

When demonstrating their approach, they reported the requirements defects distribution, classified by nature (see Figure 3.9).

Functional and Quality Based Classifiers, 1992 – 2009

Next we present defect classification taxonomies that are functional and quality based. In our research we consider that the functional classifiers represent the function of the requirement in the product (e.g. interface, performance, environment, functional).

		Bell Thayer (1976)	Basili Weiss (1981)	Ackerman *et al* (1989)	Sakthivel (1991)	Chillarege *et al* (1992)	Grady (1992)	Schneider *et al* (1992)	Porter *et al* (1995)	Hayes(03) *et al* (06)	Walia Carver (07/09)	Kalinowski *et al* (10)	#
1	Not in current baseline	1.50%											1
2	Out of scope	7.20%											1
3	Missing/Omission	21.00%	24.00%							10.80%		23.50%	4
4	Incomplete	merged		Yes	Yes					23.30%			4
5	Inadequate	merged											1
6	Incorrect	34.80%	37.00%		Yes					30.11%		35.30%	5
7	Inconsistent	9.10%	10.00%	Yes	Yes			23	Yes	13.07%	Yes	5.90%	9
8	Incompatible	merged											1
9	New	7.20%											1
10	Changed Requirement	merged											1
11	Typos/Clerical	9.90%	23.00%										2
12	Unclear	9.30%											1
13	Ambiguity		4.00%	Yes				15	Yes	13.07%	Yes	11.80%	7
14	Wrong Section/Misplaced		1.00%						Yes	1.14%	Yes		4
15	Other		1.00%								Yes	5.90%	3
16	Infeasible				Yes					0.00%			2
17	Untestable/Non-verifiable				Yes					0.00%			2
18	Redundant/Duplicate				Yes					2.27%			3
19	Missing Functionality/Feature						/u/w/c/b	34	Yes		Yes		4
20	Missing Interface					/incorrect		11	Yes		Yes		4
21	Missing Performance							7	Yes		Yes		3
22	Missing Environment							9	Yes		Yes		3
23	Missing Software Interface						/u/w/c/b						1
24	Missing Hardware Interface						/u/w/c/b						1
25	Missing User Interface						/u/w/c/b						1
26	Missing Function/Description					/incorrect	/u/w/c/b						2
27	Missing Requirement/Specification						inadequate						0
28	Missing/Incorrect Checking					Yes							1
29	Missing/Incorrect Assignment					Yes							1
30	Missing/Incorrect Timing/Serialization					inadequate							0
31	Missing/Incorrect Build/Package/Merge					inadequate							0
32	Missing/Incorrect Documentation					inadequate							0
33	Missing/Incorrect Algorithm					formal spec							0
34	Incorrect or Extra Functionality								Yes		Yes		2
35	Data Type Consistency								Yes				1
36	Over-specification									1.14%			1
37	Not Traceable									2.27%			1
38	Unachievable									0.57%			1
39	Intentional Deviation									2.27%			1
40	General										Yes		1
41	Extraneous Information											17.60%	1

For each defect classifier we indicate the authors who used it. The following information appears: Yes if we have no further information; the percentage of occurrence of a defect using the data of the experiment done with more data points; the quantity of defects; merged when the author used it merged with the classifier that is above that one; inadequate when we consider that the classifier is not useful for requirements defects; /incorrect, indicating that the authors also used the 'incorrect' prefix; /u/w/c/b indicating formal spec. (specification) when we consider that such defect classifier would only be applicable if the requirements were specified with formal language.

Figure 3.9: Defect classifier per authors by chronological order from left to right.

Schneider et al. (1992), identified two classes of requirements defects to use when reviewing user requirements documents: Missing Information and Wrong Information(Figure 3.9). In 1995, Porter et al. compared requirements inspection methods. They performed an experiment where two Software Requirements Specification (SRS) documents were inspected with a combination of *ad hoc*, checklist and scenario inspection methods. The checklist was organised in categories, resembling a defect classification: omission (missing functionality, performance, environment or interface) and commission (ambiguous or inconsistent information, incorrect or extra functionality, wrong section). The scenarios also included categories: data type consistency, incorrect functionality, ambiguity, and missing functionality. The authors concluded from their results that the scenario inspection method was the most effective for requirements.

Later, in 2007, Walia and Carver repeated an experiment to show the importance of requirements defects taxonomy. They involved software engineering students in a SRS document review using a defect checklist. The students repeated the review, after being trained in the error abstraction process. The results of the experiment showed that error abstraction leads to more defects found without losses of efficiency and the abstraction is harder when people are not involved in the elaboration of the SRS and have no contact with developers. Requirements defects were classified as: general, missing functionality, missing performance, missing interface, missing environment, ambiguous information, inconsistent information, incorrect or extra functionality, wrong section, other faults. This experiment was applied to error abstraction; we consider that a similar experiment is useful to validate defects classification.

Along the years researchers introduced classifiers to fulfil the specificities of requirements defects. Some reused existent classifications and conducted experiments to analyse the impact of different methodologies in SRS inspections. Figure 3.9 summarises the relation between authors and classifiers.

3.4 Effort Estimation

We did a literature review, designed to gather information to answer the following research questions:

- Which effort estimation methods exist?
- How can we define the effort estimation accuracy?
- Which factors are considered on effort estimation?
- Which factors affect effort estimation accuracy?

Part of our strategy was to analyse errors in schedule, effort and duration of projects and find the causes of those deviations, already identified by other researchers.

Jørgensen and Shepperd (2007), defines **estimation approach** to name the method used to estimate, which includes regression, analogy, expert judgement, work break-down, function points, classification and regression trees, simulation, neural networks, theory, Bayesian and combination of estimates. We found several effort estimation methods that other researchers classified. We merged overlapping classifications: expert based (Moløkken and Jørgensen, 2004), expert judgement/expert estimation (Lopez-Martin, 2011) and Knowledge-based (Jun and Lee, 2001); Model Based (Moløkken and Jørgensen, 2004), Statistical Model (Jun and Lee, 2001) and Algorithmic Model (Lopez-Martin, 2011); Artificial Intelligence (AI) (Jun and Lee, 2001) and Machine Learning (Lopez-Martin, 2011). The following classifications were considered:

- **Expert Based/Expert judgement/Expert Estimation/Knowledge-Based** – intuitive processes that aimed at deriving estimates based on the experience of experts on similar projects, expert consultation (Lopez-Martin, 2011). Include Intuition and experience, and Analogy by comparing completed similar tasks (Moløkken and Jørgensen, 2004);

- **Model Based/Statistical Model** – are software cost models, including formal estimation models and algorithm driven methods (Moløkken and Jørgensen, 2004); based on mathematical functions between causing factors and resulting efforts, where the estimating parameters are based on historical data (Jun and Lee, 2001; Morgenshtern et al., 2007), and linear and non-linear regression (Lopez-Martin, 2011). For example, COCOMO, Use-Case based, FPA metrics, Putnam's SLIM, Doty, TRW, Bailey&Basili;

- **Artificial Intelligence/Machine learning** – includes fuzzy logic models, neural networks, genetic programming, regression trees and case-based reasoning (Jun and Lee, 2001; Lopez-Martin, 2011);

- **Other** – Price-to-win, Capacity Related, Top-down, Bottom-up, Other (Moløkken and Jørgensen, 2004). Although, the authors consider that Top-down and Bottom-up can also be interpreted as expert judgement methods.

We compile the summary of methods we found in Table A.1 and also how they were classified (see Appendix A). From all the methods in use, the one that seems to have more accurate results is fuzzy logic, even better than particle swarm optimisation (Morgenshtern et al., 2007). These methods are good to gather estimates but explaining and varying the factors can be more complex when compared to regression models with variables without transformations. People's experience should not be neglected, in particular because in some situations expert estimates can be expected to be more accurate than formal estimation models (Morgenshtern et al., 2007). Even TSP recommendeds teams to use expert judgement when there is no prior TSP data available and the guidelines are not applicable to the project (Humphrey, 2006).

We also analysed the methods used to evaluate and compare the accuracy of the estimation methods. We present their equations on the next paragraphs.

$$MeanMagnitudeRelativeError: \quad MMRE = \frac{1}{n}\sum_{i=1}^{i=n}\frac{|Act_i - Est_i|}{Act_i} \tag{3.1}$$

In 3.1 MMRE is the Mean MRE, in a given project *i*, where *n* is the number of samples, *Est* is the estimated and *Act* is the actual (Conte et al., 1986). The metric is better when lower. The MRE penalises overestimation more than underestimation (Jørgensen, 2004). The Relative Error *RE* is the fraction of 3.1, without the module, showing the direction of the estimate.

$$PercentageofPredictors(r): \quad PRED(r) = \frac{k}{n} \tag{3.2}$$

In the PRED(r) equation (3.2) *k* is the number of projects in a set of *n* that whose MRE <= *r*, that is, fall within r of the actual value. A cost model is considered accurate when MMRE is at most 0.25 and PRED(25) is at least 75% (Braga et al., 2008).

Jørgensen (2007) used 3.3 to determine estimation accuracy as follows:

$$EstimationAccuracy: \quad z = \frac{Est}{Act} \tag{3.3}$$

$$MagnitudeofErrorRelative: \quad MER_i = \frac{|Act_i - Est_i|}{Est_i} \tag{3.4}$$

MER (3.4) is calculated for each observation *i* whose effort is predicted, and its aggregation over multiple observations gives the Mean MER (MMER) (Smith et al., 2001; Lopez-Martin, 2011; Hari and Prasad Reddy, 2011).

Other equations referenced by Hari and Prasad Reddy (2011) are the Variance Accounted-For, equation 3.5, Mean Absolute Relative Error, equation 3.6, and Variance Absolute Relative Error, equation 3.7.

$$VarianceAccountedFor: \quad VAF = 1 - \frac{var(Act - Est)}{var(Act)} \tag{3.5}$$

$$MeanAbsoluteRelativeError: \quad MARE = mean\frac{|Act - Est|}{Act} \tag{3.6}$$

$$VarianceAbsoluteRelativeError: \quad VARE = var\frac{|Act - Est|}{(Act)} \tag{3.7}$$

Statistic analysis of the models residuals was also used by van Koten and Gray (2005), including the Standard Deviation of residual error (3.8).

$$StandardDeviation: \quad SD = \sqrt{\frac{\sum(Act - Est)^2}{n-1}} \tag{3.8}$$

From the analysis of all the variables used by other researchers, when analysing effort estimation accuracy, we found that statistic methods would be more appropriate, however other authors compared them to MMRE and PRED and concluded that they did not lead to a significant improvement (Jørgensen, 2004). From the variables more commonly used in effort estimation, MER is preferable to MMRE, because MER measures the inaccuracy relative to the estimate (Foss et al., 2003). Therefore, we used it in our model to evaluate the quality of implementation of the practice "Estimate Effort" (see 5.3 Evaluation of the Estimation Process).

We also analysed the factors influencing the effort estimation accuracy, grouped as factors to be considered on the:

- **Estimation Process** - including experience and skills of estimators and executors; complexity of products, processes, tools; schedule, time constraints; level of detail; uncertainty of project, technology; personnel availability, turnover; support tools; people localisation; cumulative complexity; methodology, development phases; customer involvement; internal and external communication;
- **Project Execution** - including project execution, priorities, information completeness; personnel availability and turnover; progress and status control; risk management; reporting.

We include the tables detailing the factors of the effort estimation process and the factors of the development process in appendix A. The factors raised were considered when building EQualPI's Data Dictionary and defining the Domain Model (4.6.2 Domain Model). In fact, factors that influence effort and duration estimation were studied and tested for years, Morgenshtern et al. (2007) compiled them:

H1– Estimators in charge of task bias the estimates, (DeMarco, 1982).

This is possibly explained by the planning illusion, (Buehler et al., 1994). Self estimates are underestimated (DeMarco, 1982), and so are the tasks that are perceived as easy, while tasks that are perceived as difficult are overestimated.

H2– Large variance in time and budget lead to estimates below actual values (Hihn and Habib-agahi, 1991).

H3– Project size affects time estimations.

The justification is given by Brooks (1982), saying that "In small projects communication is tighter and problems are easily dealt with".

H4– "Uncertainty reduces estimation accuracy and hence the project performance" (Nidumola, 1995).

H5– Complex tasks insufficiently broken down lead to underestimation errors (Hill et al., 2000).

Estimation errors are independent and breaking the project into smaller work packages reduces estimation statistical errors (Raz and Globerson, 1998). Morgenshtern et al. (2007) confirmed this hypothesis.

H6– Managerial control improves estimation accuracy (Van de Ven and Ferry, 2006).

H7– Good relation between developers and the customer have a positive effect on estimation accuracy (Van de Ven and Ferry, 2006).

H8– Sense of responsibility and commitment contribute to estimation accuracy.

H9– External estimators provide lower estimates than the developers (Lederer et al., 1990).

Morgenshtern et al. (2007) considered that the duration of the project and effort estimation errors were affected by uncertainty in the project, effort invested in the estimation process and in managing the estimates, and the estimators experience. Therefore, they tested the following hypotheses:

H10 – "Projects with higher level of uncertainty exhibit larger duration and effort estimation errors."

They defined project uncertainty as a function of other variables (see equation 3.9).

$$\begin{gathered} ProjectUncertainty = f(Duration, ManagerialComplexity, InnovativenessOfNeed, \\ TeamExperience, ResourcesAvailability) \end{gathered} \tag{3.9}$$

Where managerial complexity is a function of other variables as well, as defined in equation 3.10.

$$Managerialcomplexity = f(TeamSize, NumberOfUsers, ImplementationComplexity) \tag{3.10}$$

H10 was partially confirmed. The authors showed that project size (equation 3.11) increases duration and effort errors.

$$Projectsize = f(Duration, Effort, NumberOfUsers) \tag{3.11}$$

H10A – Innovativeness of need.

H10B – Complexity.

Both H10A and H10B tests showed that the variables increased duration and effort errors.

H10C – Estimated Duration.

H10D – Implementation Complexity.

The tests of the hypotheses H10C and H10D showed that for larger and complex projects the duration estimation errors were smaller than the overall duration, and when the team experience was lower both duration and estimation errors increased.

H11 – "Projects with higher use of estimation development processes exhibit smaller duration and effort estimation errors."

Where estimation development was defined as a function of the variables expressed in equation 3.12:

$$EstimationDevelopment = f(Purpose, WBS, EstimationTechniques, CommitmentProcess) \tag{3.12}$$

This hypothesis was partially confirmed, as the number of Work Breakdown Structure (WBS) levels was uncorrelated with estimation errors, but shorter activity durations and smaller task efforts gave smaller effort estimation errors. In Scrum and TSP is recommended that the tasks duration is kept short, e.g. around 10 hours or less (Over, 2010), if necessary they get broken into more, of smaller effort, in order to keep better control of the plan and fitting them into the development cycle duration.

Morgenshtern et al. (2007) also showed that better estimation goals definition resulted in smaller estimation errors, although with greater impact on duration than on effort, and better

estimation techniques and commitment processes resulted in smaller duration errors, but not significantly.
H12 – "Projects with higher use of estimation management processes exhibit smaller duration and effort estimation errors." (see equation 3.13)

$$\begin{split} EstimationManagement = f(&ControlOfActualPerformanceAgainstEstimates,\\ &RePlan, RiskAssessment, PromoteCommitmentToThePlan) \end{split} \tag{3.13}$$

This hypothesis was partially confirmed.
H12A – Higher customer control.

Reduces relative Effort estimation errors.
H12B – Frequent reporting.
H12C – Team performance assessment.
H12D – Risk management.

H12B to H12D reduce duration errors, additionally, H12B leads to higher commitment which results in better estimations.
H12E – Frequent work plan updates.

H12E led to better duration estimation.
H12F – Customer control (via steering committee).

Was shown to reduce relative effort estimation errors. However it was more correlated to duration due to the focus on duration management, rather than effort management.
H13 – "Projects where the estimates were made by more experienced estimators will have smaller duration and effort estimation errors." (see 3.14)

$$EstimatorExperience = f(YearsOfExperience, NumberOfProjects) \tag{3.14}$$

This hypothesis was partially supported.
H13A – Higher number of projects in the specific application.

Resulted in lower duration error.
H13B – Higher number of years of experience.

Contributed to lower duration error (but not significantly).
H13C – Estimators trained in IT project management.

Was shown that those estimators had smaller errors than the untrained ones.
H13D – Training in estimation techniques.

Was shown to improve duration estimates.

This literature review served as a base for our research, in particular when designing the experiment, analysed data and implemented the Effort Estimation Accuracy model (5.3 Evaluation of the Estimation Process). We also added hypotheses to the aforementioned ones, when we considered these relevant to EQualPI, testing part of them.

To evaluate the quality of implementation of the CMMI practices, or any process improvement in general, it is necessary to go beyond compliance as done in SCAMPI, which just partially tackles performance evaluation in a couple of PAs, CAR and OPM (CMU/SEI, 2008). It is necessary to evaluate the quality of the outcome of a practice/process, evaluating efficiency and effectiveness. Considering the challenges and work already done by other researchers who were focused on CMMI, SCAMPI, Metrics Definitions, Metrics Repositories, Process Improvements Evaluations, and Performance Measurement, we defined the EQualPI Framework. Due to the number of CMMI Practices, we chose to demonstrate it in the evaluation of PP SP1.4 "Estimate Effort and Cost." We present our Framework in the next chapter.

Chapter 4

The EQualPI Framework

In this chapter we present the EQualPI framework. To design its Metamodel we based ourselves in the SPEM 2.0 (OMG, 2008), CMMI architecture, measurement principles and the necessary alignment that must exist between the organisation (quantitative) business goals and the performance indicators. EQualPI is composed of a **Metamodel** that defines the rules and relations between its elements, a **Repository** that includes the mathematical models and data to perform a quantitative Evaluation of the implemented practices and a set of **Procedures**, including the steps to prepare EQualPI to be used in an organisation.

4.1 Framework Overview

When organisations use CMMI, their processes are aligned with the process areas they use and they implement the corresponding practices using operational practices, of which TSP is an example Phillips (2010a). In our research we term the definitions of the operational practices **methods**. We evaluate quality of implementation by determining the degree of support of the methods used by the organisation in the definition of their processes to the CMMI practices and use performance indicators to both measure the quality of implementation and the organisation performance. In this context we define the following concepts:

- **Method** - good practices, procedures, techniques, etc., that define how the work is done and support doing it. Methods are used to achieve a certain work objective.
- **Process** - includes a set of reusable elements, the methods, which can be optional, alternative or mandatory, and are used to perform work. The processes are tailorable, meaning that some methods or activities are optional and others can have alternative ones, decided according with the implied needs of the work which is to be performed.
- **Quality of Implementation** (of a practice) - refers to the way that the work related to a practice is performed. One way of working is considered better than another if it consistently produces better results in a more effective way, while keeping the cost constant or at a

reasonable value, and, consequently, spending the same or similar amount of time. We characterise the quality of implementation of a CMMI practice by a combination of **efficiency** and **effectiveness** of implementation, on one hand, and **compliance** of implementation on the other (i.e., alignment either with CMMI recommendations or with what is prescribed by the concrete implementation method used), all measured by appropriate **performance indicators** (PIs), which may possibly be dependent on the practice and implementation method used. By considering these three quality characteristics, we are looking both at **how** the work is done and **what** its performance results are.

- **Performance Indicators** - derived metrics that measure the organisation performance and/or the quality of implementation. Their aggregation indicates the degree of institutionalisation of the practices necessary to achieve generic goals and high maturity, and consequently to allow practices' evaluation.

- **Controllable factors** - methods that are distinguishable, enumerable, reusable, in some cases quantifiable, and which can be acted on or influenced by the organisation, business unit or team. Those factors can be related to each other, or not (Stoddard et al., 2009).

- **Uncontrollable factors** - are the circumstances, environment variables, external factors that the organisation, business unit or project team cannot change and therefore must accommodate (Stoddard et al., 2009).

To build the EQualPI we followed a bottom-up approach, as represented in Figure 4.1, based on the principle that the quality of the practices implemented in lower maturity levels is reflected in the practices of higher maturity levels. The practices used individually, in each team, project, organisation unit and in the organisation give the implementation quality as a whole. The result of the evaluation is signalled by a colour scheme (red, yellow and green), defined by the threshold of the performance indicator.

To be compliant with the CMMI model the organisation needs to implement the SGs and Generic Goals (GG) enounced in the model, in the current maturity level and all the precedent ones (Chrissis et al., 2011). To satisfy a goal, the Generic Practices and Specific Practices or acceptable alternatives to them need to be fulfilled.

Our approach to the problem is represented in Figure 4.2. For each Specific Goal and/or Specific Practice we identified the methods that are documented in the literature and the ones that are used in the organisations to implement the CMMI goal or practice. The organisation's processes and tools are documented in the Quality Management System.

The practices implementation is reflected in the implementation of the generic goals. We map the methods with the CMMI practices and/or goals, and evaluate the quality of implementation. That quality is related to the percentage of the CMMI practice that the method covers and the way it is implemented. The SCAMPI evaluates if a practice is fully implemented, partially implemented or not implemented and in certain cases the evidence of a practice is a documented process and records of its usage, and collateral evidences of its application (refer to 2.4 CMMI Architecture and

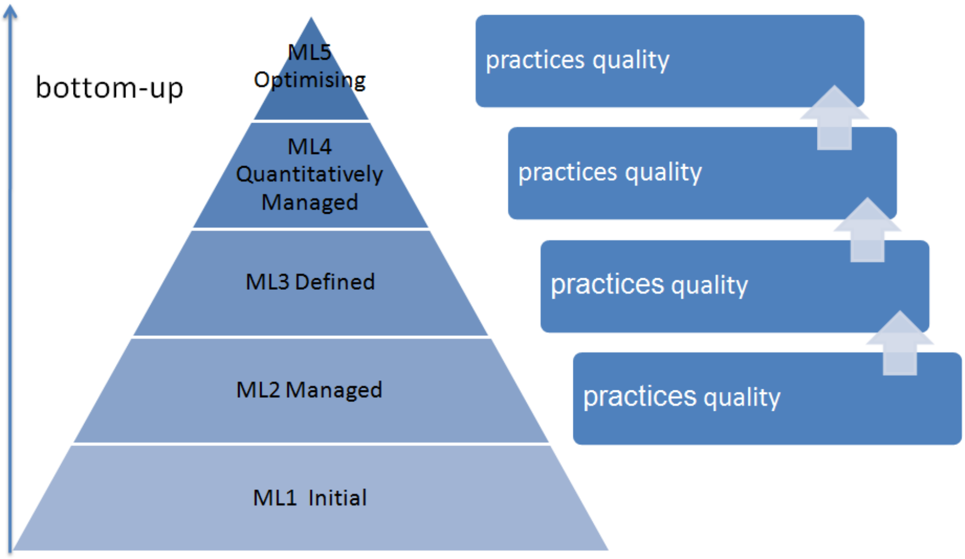


Figure 4.1: Bottom-up evaluation of practices implementation.

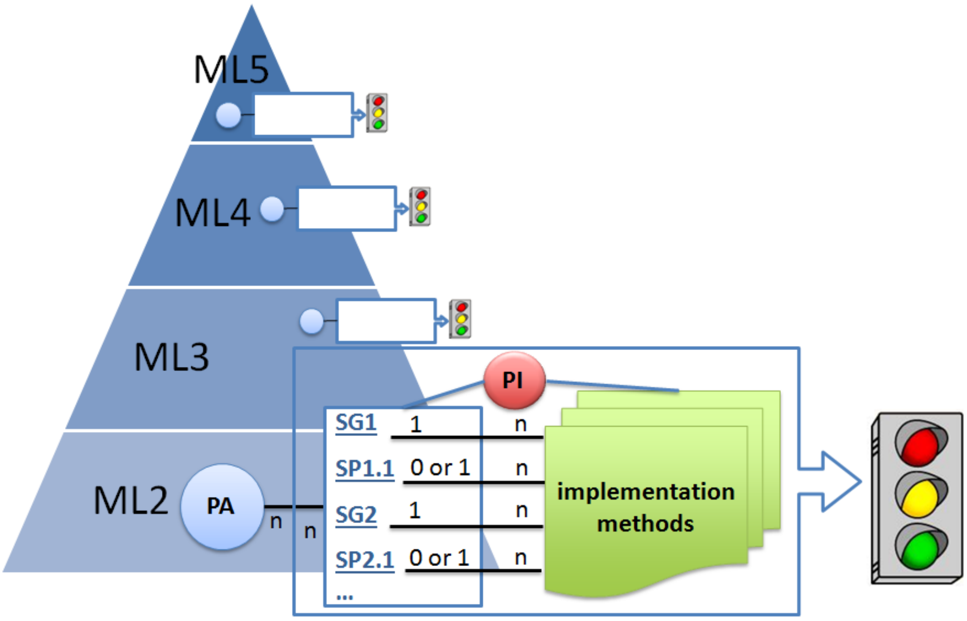


Figure 4.2: Building the Evaluation Framework (Lopes Margarido et al., 2011b).

Legend: ML – Maturity Level, PA – Process Area, SG – Specific Goal, SP – Specific Practice, n – one or more, PI – performance indicator.

Appraisal Method). The effectiveness of the methods application is not considered. To analyse the practice implementation we based the evaluation on the recommendations concerning the method that is to be found in the literature.

The percentage of usage of a practice is reflected in the generic practices. For example, if 100% of the projects use the process, tailored from the organisation's set of standard processes, then the process is institutionalised as a standard process and "GG3 – Institutionalise a Defined Process" is fulfilled. We consider that the percentage of usage of a method in the organisation can be used to evaluate GG and higher maturity practices.

The purpose of the research is to develop a framework that allows organisations to select and adapt methods that improve the quality of implementation of CMMI practices, allowing them to monitor practices performance and anticipate the impact of changing their processes in the performance of those practices. Our goal is to help improve and replicate success by identifying the elements in the ways of doing work (controllable factors) that have most impact in the efficacy and efficiency of the processes. Those elements are related to the methods used in the execution of the process. The Framework is composed of a Metamodel, shaping a Repository of performance indicators, to evaluate the quality of implementation of CMMI practices, dependent on the methods used to implement those practices. The performance indicators are tailorable, defined as mandatory or optional, and mapped with profiles according to maturity level and the methods of the organisation. Additionally, the Framework includes procedures for setup (tailoring), use in practice to evaluate quality of implementation and do process improvements, and supporting choice of indicators. However, CMMI has 22 Process Areas, with the respective Specific Goals and several Specific Practices, and maturity levels 2 and 3 have Generic Goals, with their corresponding Specific Goals. Developing a framework to include the entire model in a single Ph.D. research would be infeasible. Consequently, we decided to demonstrate our theory by analysing the factors that influence the quality of implementation of the CMMI Project Planning SP 1.4 "Estimate Effort and Cost".

In my experience, the effect of having a project plan that was built solely based on the project's proposal, without a review once the engineering team was set, lead to poor project execution, scope misalignment and overhead. We consider that the results of project estimation are determinant in projects' success. For example, with good effort estimation the uncertainty in the plan is lower, and the team is less stressed and subject to extra unconsidered tasks that should have been considered as part of the scope of the project, but forgotten. We also believe that it would be easier to determine how the Framework can be extended by demonstrating it in this particular area because we think that PP SP1.4 depends on the quality of other CMMI practices, including the ones used in the development process. We focused on effort estimation and left cost out of the research because effort estimates are already a dominant factor considered in cost estimation.

The quality of a project plan depends on the quality of estimates and some of the factors that influence those estimates can and need to be controlled. We focused our research on the factors that we can control, in order to do better Effort Estimation, i.e. with indicators that are drivers of the results. Those indicators are the ones related to the Effort Estimation process and they show how well the process is defined and executed. Other factors, in particular the ones related to the project execution itself were monitored with two purposes: understand the percentage of the effect that they have on the Effort Estimation Accuracy and to characterise our datasets, so that the research can be reproducible.

Our goal was to demonstrate that it is possible to evaluate the quality of implementation of the CMMI practices based on the performance resultant from the methods used in their implementation. Such evaluation shall support organisations to achieve the benefits of the model and anticipate the impact of changes in their processes.

We consider that a performance indicator that shows the quality of effort estimates is the Effort Estimation Accuracy, based on the deviation between the estimated and actual effort. The effort estimation deviation depends on several factors, such as:

- Definition of the estimation process;
- Execution of the estimation process;
- Execution of the project.

These factors contribute to the quality of implementation of the Effort Estimation Accuracy, also designated as construct or dependent variable. We determined the percentage of the estimation accuracy that is affected by the outputs of the estimation process – quality of the process and its execution – and focus on the analysis of factors that contribute to that percentage. Therefore other external factors, related to the execution of the project and the team, had to be controlled and recorded but are outwith the scope of the research. In practice, we formulate the problem as follows: the dependent variable Effort Estimation Accuracy, Y, is a function of n controllable factors X_c and i uncontrollable factors X_{nc} (see equation 4.1).

$$DependentVariableDefinition: \quad Y = f(X_{c1}, X_{c2}, ...X_{cn}, X_{nc1}, X_{nc2}, ...X_{nci}) \tag{4.1}$$

Regarding the performance indicators, we do understand that some of them depend on the implementation methods. However, we wanted them to be good predictors of the performance of the organisation regarding **Effort Estimation Accuracy** and to be usable. We consider that usable performance indicators are the ones that bring added value and can be collected in a cost effective manner (without high costs and overhead). They allow the organisations to improve their performance by following the guidelines of implementation of the methods, knowing current and desired performance and knowing impact of process changes on performance indicators.

4.2 EQualPI's Architecture

In this section we present the EQualPI framework complemented with technical description.

In Figure 4.3 we present the architecture level 0, the deployment perspective, of the EQualPI framework. This perspective is used to give the physical context in which EQualPI will operate, within an organisation context. We also identify external agents that communicate with the Framework.

In the company work environment and while people do their work, metrics are collected from **Workstations** to the **Company Database**, where the data on those metrics are stored. The

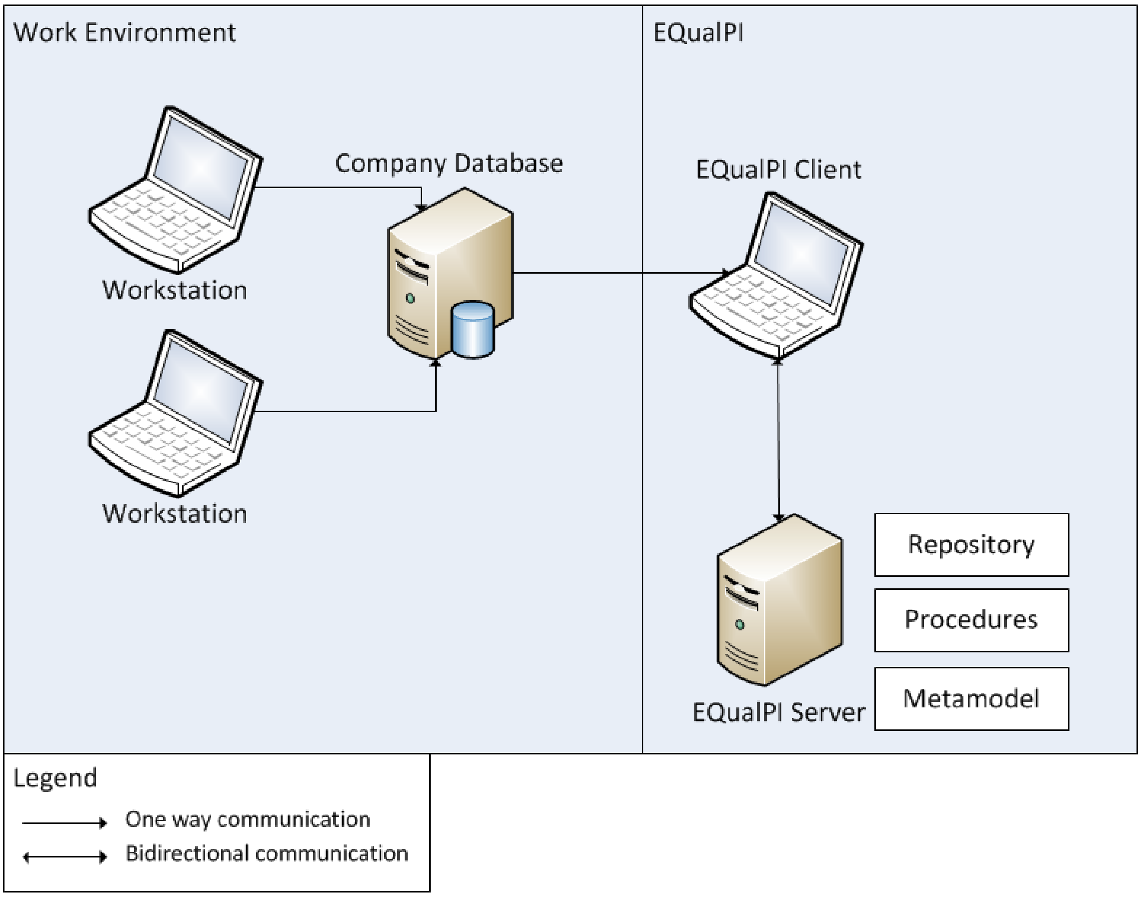


Figure 4.3: EQualPI architecture level 0 - deployment perspective.

EQualPI framework is composed of a **EQualPI Client**, that presents the User Interface (UI), which allows the user to access the **EQualPI Server** where evaluations are performed. The server includes the **Repository** and **Procedures**, both shaped by the **Metamodel**. The performance indicators to evaluate the quality of implementation of the CMMI practices are stored in the Repository. They may depend on the methods used to implement those practices and on the business goals. For that reason the performance indicators are tailorable, defined as mandatory or optional, and mapped with profiles according with maturity level and methods used by the organisation. Furthermore, organisations can add metrics that are not yet in the system, but they must be aligned with business goals, methods and CMMI practices and respect the Data Dictionary rules. Additionally, EQualPI includes procedures for setup (tailoring), use in practice and support the choice of indicators.

We used a metamodel to define our Framework. According with Álvarez et al. (2001), in a model architecture, a model at one layer specifies the models in the layer below and it is viewed as an instance of some model in the layer above: "The four layers are the meta-metamodel layer (M3), the metamodel layer (M2), the user model layer (M1) and the user object layer (M0)." In our case we represent the metamodel layer M2 and the user model layer M1. Figure 4.4 presents the level 1 of EQualPI's architecture, the static perspective, including the layers of the framework and the metamodel. This perspective represents the modules that are part of EQualPI, which is then implemented in a three-tier architecture.

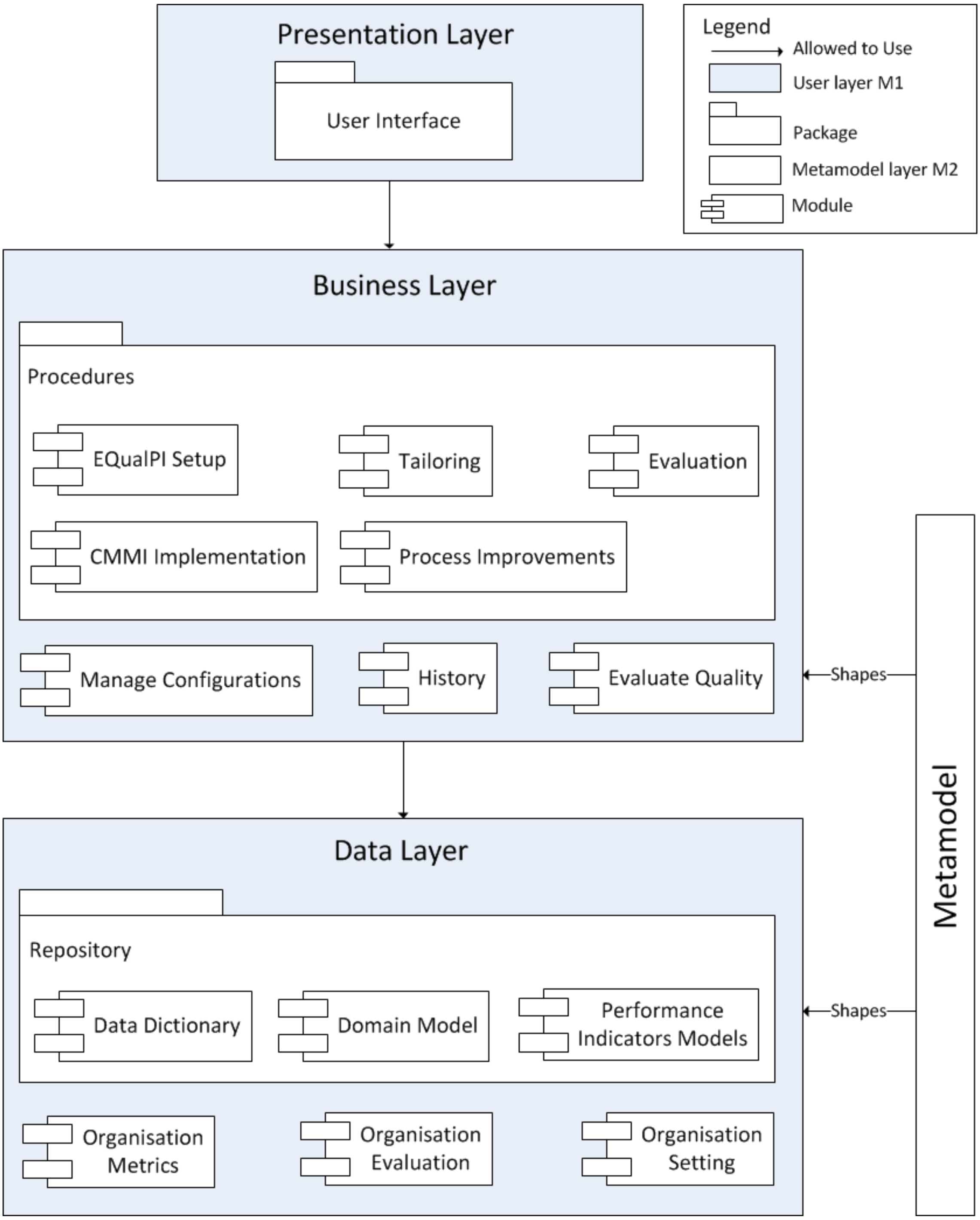


Figure 4.4: EQualPI architecture level 1 - static perspective.

EQualPI has three layers: **Presentation**, **Business** and **Data**. The Presentation Layer includes the **User Interface** that can only communicate with the **Business Layer**. The UI formats and presents the information to the end user. Through it, the user sends the information that is necessary to setup the Framework and the data of the organisation, formatted according with the Data Dictionary. To evaluate quality, the user inserts the parameters necessary to do it; EQUalPI presents the results of the evaluation in the UI. The user can also send parameters to consult previous evaluations, whose results are presented in the UI as well.

The Business Layer includes all Procedures that are necessary to use EQualPI and include:

- **EQualPI Setup**, how to align business goals with performance indicators and practices. The information that needs to be provided to start using the Framework.
- **Tailoring**, with instructions to select the practices, methods and performance indicators the organisation will use.
- **Evaluation**, that explains how the evaluation and aggregation are done.
- **CMMI Implementation**, that includes guidelines and a checklist to help implement CMMI avoiding problems that often occur when implementing the model.
- **Process Improvements**, the steps to do process improvements and how to use the Framework to evaluate them.

The Business Layer allows to **Setup (the) Framework**, receiving the data of the organisation and preparing EQualPI to work with those data. The organisation configurations follow the rules defined in the module **Manage Configurations**. It is also possible to consult the **History** of the organisation, such as previous evaluations and settings, and actually evaluate the quality of implementation or of improvements using the **Evaluate Quality** module.

In the Data Layer, only accessed by the Business Layer, all data are stored, including the data of the organisation and the data that comes with EQualPI, belonging to the Repository. The Repository holds the **Data Dictionary**, which describes all variables properties and all base and derived measures that allow to evaluate CMMI practices. The **Domain Model** explains the relation between variables that are specified in the Data Dictionary. Finally, the Repository also includes the **Performance Indicators Models** that are the ones that actually evaluate the quality of implementation of a given practice. The organisation data are composed of **Organisation Metrics**, **Organisation Evaluation**, which includes all evaluations and the data of the existing configurations, and **Organisation Settings** that is the structure of the processes implemented by the corresponding methods and evaluated by the respective PI. Such structure defines the practices and methods in use, mapped with the organisation goals. The settings history is also stored, so if an organisation needs to see an old evaluation it can also see the settings that were used then. The **Metamodel** shapes the Business and Data Layers, as they follow the definitions it contains.

4.3 EQualPI's Metamodel: Repository and Evaluation

A metamodel can support the design and implementation of a framework and establishes dependencies and rules necessary to use it in practice. In fact, it describes and analyses relations between concepts. We represent the Metamodel of the **Repository** in Figure 4.5 and of the **Evaluation** in Figure 4.6.

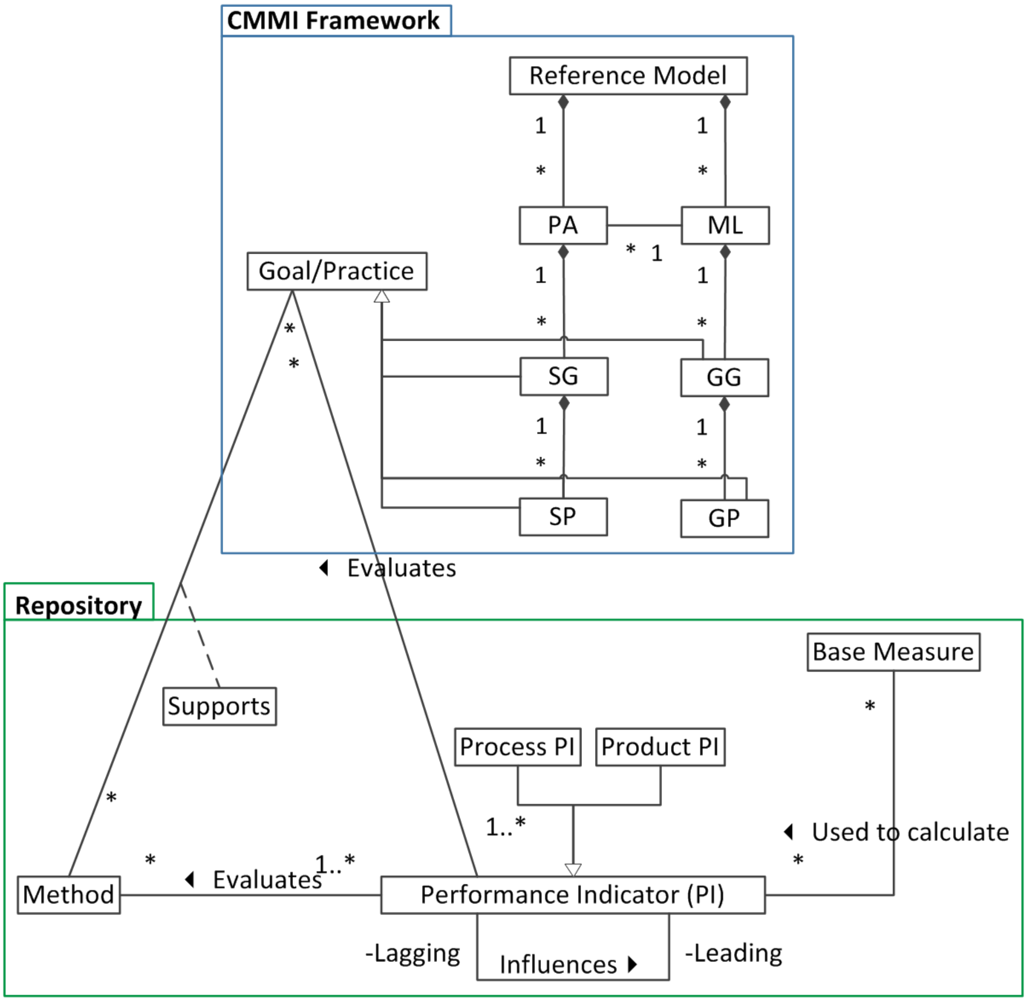


Figure 4.5: EQualPI's Repository Metamodel (Lopes Margarido et al., 2012).

Legend: PA - Process Area, ML - Maturity Level, SG - Specific Goal, SP - Specific Practice, GG - Generic Goal, GP - Generic Practice, PI - Performance Indicator.

Regarding Figure 4.5, the organisation selects the constellation of the CMMI framework to be implemented, in the figure designated as **Reference Model**. According to the organisation goals, the practices to be implemented are selected, which together may allow the achievement of a maturity level. The organisation goals are aligned with performance indicators and methods that support them and the CMMI practices that are being implemented. The **Performance Indicators** allow the evaluation of practices and are derived measures, calculated through **Base Measures** that

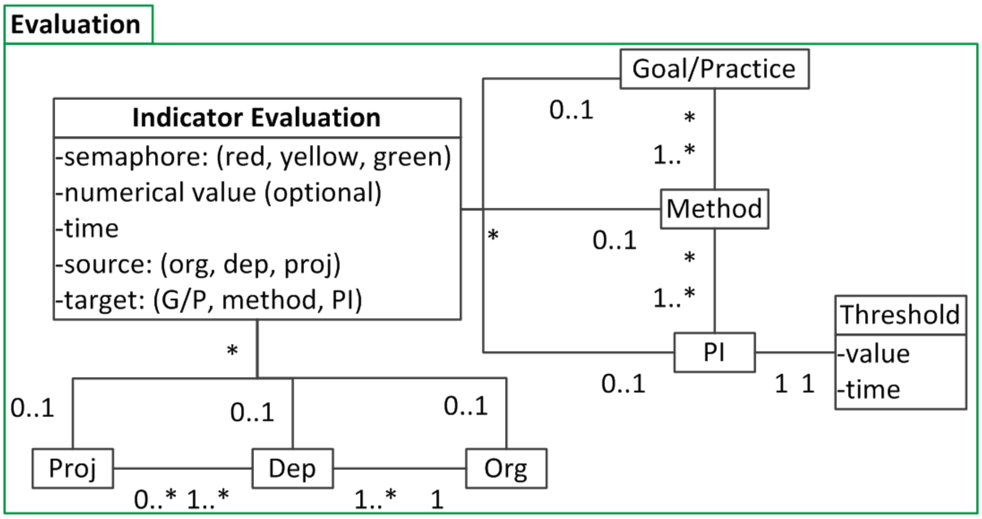


Figure 4.6: EQualPI's Evaluation Metamodel (Lopes Margarido et al., 2012).

Legend: Proj - Project, Dep - Department, Org - Organisation, PI - Performance Indicator, G/P - Goal or Practice.

the organisation collects through time. There are **Process PI** evaluating the process and **Product PI** that are related to the product. **Leading** indicators, helping to predict the outcome of the work done following a given method, and **Lagging** indicators, used to appraise the process and product implementation performance. A lagging indicator in a phase of the project lifecycle may be leading indicator in the next phase.

For instance, assume that we want to evaluate the quality of implementation of practice "SP2.2 Conduct Peer Reviews" of the Verification process area and that reviewing follows two TSP guidelines: use checklists derived from historical data, and review at a moderate pace. Here, one can measure **efficacy** by *review yield* (percentage of defects detected), **efficiency** by *defect detection rate* (defects detected per hour), and **compliance** by *checklist usage* (a qualitative PI with values, *not used*, *ad-hoc checklist*, and *checklist derived from historical data*) and *review rate* (size reviewed per hour), compared with some recommended values.

A rich set of PI usually combines process and product indicators, and leading and lagging indicators. In the given example, *review yield* is a lagging performance indicator, as the remaining defects can only be known *a posteriori*. Compliance indicators are often leading indicators; they influence and can be used to predict and control the values of lagging indicators. In the example, *review rate* is commonly considered a leading indicator of the *review yield* in TSP literature. The *density of defects* found in a review is a product performance indicator, whilst the *review rate* is clearly a process performance indicator.

To conduct an evaluation (see Figure 4.6) the **Goals/Practices** are mapped with one or more **Methods** that implement them and the **PIs** used to evaluate those methods and, consequently, the Practices. A PI has a **Threshold** that at a given **time** has a particular **value** and is established by the organisation from the analysis of what is considered to be the limit of the normal behaviour for that indicator. Thresholds have different levels, used to determine the PI **semaphore** colour (red, yellow, green), established according with the organisation quantitative business goals and processes baselines, and define its normal behaviour regarding a PI. The evaluation of a method can be done by aggregating the result of the evaluation of one or more PIs; similarly the evaluation of a practice is given by the aggregation of the evaluation results of their implementing methods. Note that a practice can be implemented by a group of methods; therefore a method can be **mandatory**, **alternative** or **optional**.

The PI can be collected at a given **Source**, a **Project**, a **Department** or in the **Organisation**. Moreover, the evaluation of the several projects can be aggregated to evaluate a department, and the results of a department can be aggregated to evaluate the entire organisation. In any of these cases what is evaluated may be a PI, a method or a practice. This approach also helps monitoring of whether the organisation quantitative goals are achieved.

There are three dimensions of aggregation of evaluation results, the already mentioned **target** (practice, method, PI) and **source** (project, department and organisation), to which we add **time**. Aggregation in time is done by analysing the organisation's data in a selected period, given the methods and thresholds at that moment. Aggregation at organisation level indicates the degree of institutionalisation of the practices necessary to achieve generic goals and high maturity, and consequently, allow their evaluation. A project, department or organisation can also use target aggregation to evaluate a method or a CMMI goal/practice. The evaluation by aggregation of colours is done as follows:

- Green – all green;
- Yellow – at least one yellow and no reds;
- Red – at least one red.

We are aware that results aggregation can be complex and not simple sum or median of PIs' evaluations, but the aggregation at source level is outwith the scope of this research. Nonetheless, the thresholds of the organisation may both be rigid to the point that for a matter of decimal places the PI target is not reached, depends on the tolerance defined.

4.4 Procedures Package

The Business Layer's package **Procedures** (see Figure 4.7) and its modules give organisations the necessary information to use the Framework in practice. Procedures include:

- Instructions to deploy the Framework and align it with the organisations practices and goals;
- Checklist to support the CMMI implementation;
- Instructions to populate the EQualPI database and calibrate the models;
- Instructions to evaluate the implemented practices;

- Instructions to conduct processes improvements pilots and deploy them.

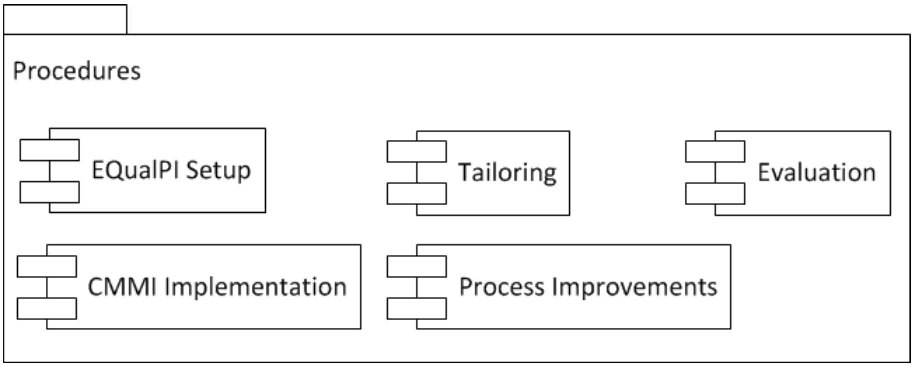


Figure 4.7: Contents of the Procedures package.

4.4.1 EQualPI Setup, Tailoring and Evaluation

The preparation of the Framework by an organisation, or setup, is done following the Repository Metamodel that we presented in Figure 4.5. The organisation must choose a reference model in the CMMI framework; it may wish to implement a process area or a maturity level, or it may implement just a subset of process areas that do not allow achieving a capability or maturity level but are relevant for the organisation. The organisation may choose to implement all specific practices to achieve a goal, or it may indicate alternative practices. Similar choices can be made in the case of implementing a generic goal, by following the generic practices or following alternative ones. After this, the organisation must indicate the methods used to implement the goal or practice, and the performance indicators to evaluate them. The methods are selected from a pool of methods existent in the EQualPI Repository along with the corresponding performance indicators, but they may also be added by the organisation. EQualPI allows implementation according with the CMMI model; the user just indicates the level and representation and all information is automatically loaded. The steps of the setup are represented in Figure 4.8.

After EQualPI is configured, and practices are mapped with methods and performance indicators, the organisation needs to setup thresholds and quantitative goals per Performance Indicator. The data used for that purpose, in case of not having historical data, can be a benchmark from the industry. Depending on the indicator, the thresholds define performance intervals of acceptable (green), alarming (yellow) or unacceptable (red) behaviour, triggered by reaching a given value or going out of a range of values.

To populate EQualPI's database with **Organisation Metrics**, the organisation exports information of the company database in the format of the Data Dictionary. When an organisation carries out an evaluation the database needs be updated until the end date that the evaluation refers to. The organisation may also choose the source (organisation, departement, project) and target (performance indicator, method, practice) of the evaluation, as explained in 4.2 EQualPI's Architecture.

To setup the Framework an organisation must follow a set of steps according with the flowchart:

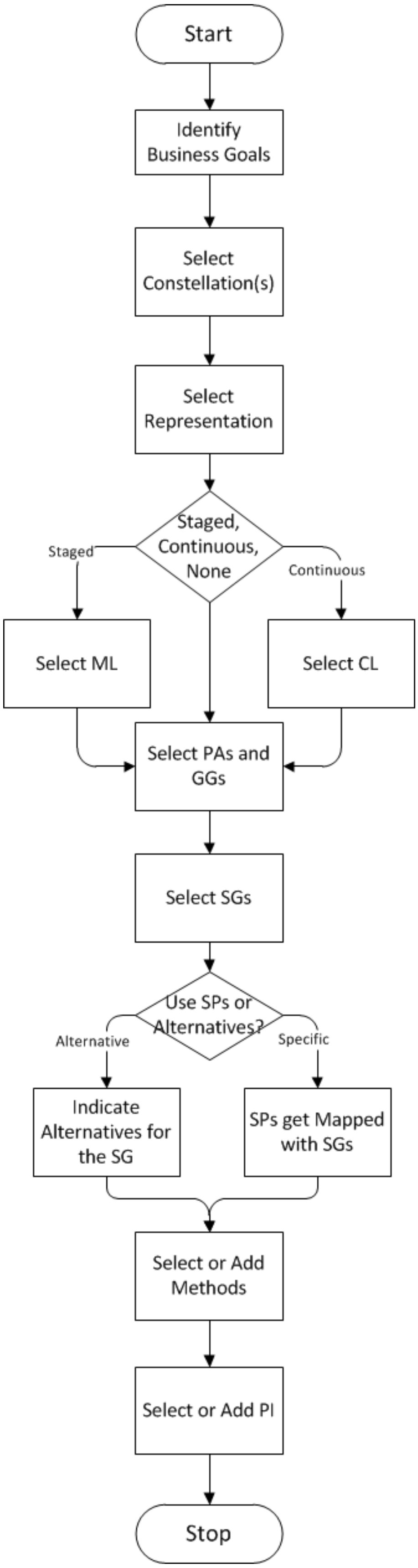


Figure 4.8: Flowchart of the setup of the EQualPI framework.

1. Identify business goals;
2. Select the Reference Model (CMMI constelation(s)) to use;
3. Select the model representation (staged, continuous or none);
4. If staged was selected, select the Process Areas and Generic Goals to implement;
5. If continuous was selected, select the Maturity Level or the Process Area;
6. If none was selected select the Capability Level or the Process Areas to implement;
7. Select the Specific Goals;
8. The Framework suggests the use of Specific Practices or the user may use Alternative Practices;
9. If Alternative Practices were selected the user must indicate them and map them with the Specific Goal;
10. If Specific Practices were selected they will be mapped under the corresponding Specific Goals;
11. Per practice or alternative practice select the method(s) and indicate which ones are mandatory, alternative and optional;
12. If a method is not in the system add it and map it with the practice it implements;
13. Per method EQualPI suggests a list of Performance Indicators that are in the pool, according with the maturity profile (in case the PAs selected do not allow to achieve a level, then the Performance Indicators suggested indicate the maturity profile). Select the performance indicators;
14. If the desired performance indicator does not exist, specify it according with the Metamodel and Data Dictionary, and map it with the method.

In practice, when an organisation uses the Framework, it tailors the CMMI process areas/ practices, methods, and corresponding performance indicators, that will be used. The organisation executes the evaluation process, after defining the thresholds for the performance indicators according with its business goals. From executing an evaluation the organisation will get a colour for the quality of implementation of the CMMI practices, organisation performance and impact of performance improvements (in case it changed its processes). The performance indicators are based on the business goals, which may be financial, marketing, quality and related to the customer. Those goals are drilled down to departments and projects goals. Having goals related to better planning, faster development, customer satisfaction, etc., we can map them with performance indicators that show us the predictability, productivity, re-work, defects delivered, etc.

An example of evaluation is depicted in Figure 4.9. The analyst or stakeholder would select a PI to analyse in a period of time. In the example given the PI is *Schedule Estimation Error*. When analysing a project the person verifies in which range of values the data point lies (in case there are upper and lower thresholds) and analyse the colour of the evaluation accordingly. The evaluation of a department would be given by aggregating the results of the department's projects. In the example, department 1 (D1) has three projects (P1, P3 and P4) and one is red, so it is evaluated as red; D2 has a single project that is yellow, so the result for that department is yellow; and D3 has one project P5 that is green, so the department evaluation is green. When evaluating the organisation, D1 was evaluated as red so the organisation is red.

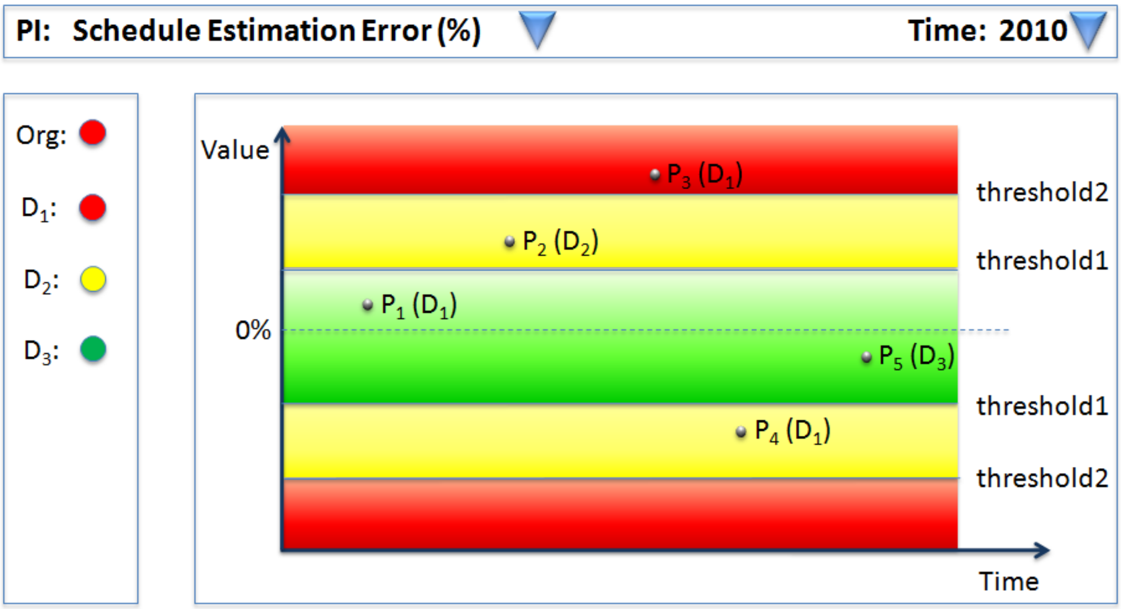


Figure 4.9: Evaluation of the Schedule Estimation Error (Lopes Margarido et al., 2011b).

Legend: PI - Performance Indicator, Org - organisation, D1 - department 1, P1 - project 1.

Note that we are giving a simplified presentation of the evaluation; aggregation of results is more complex and diverse. Organisations can establish their quantitative business goals giving a certain margin so that all departments can achieve the intended results. In that case the departments have a smaller margin, to make sure the organisations goals are achieved, and projects have even smaller margins to make sure that departments' goals are achieved. This means that a project could be yellow and the department would still be green, or a department could be yellow and the organisation would still be green. There are also cases of aggregation that requires considering different variables at different organisation levels. The PIs may have different formulas or base measures at different organisation levels, not being reduced to a sum of parts. When defining the aggregation rules at different levels of granularity it is necessary to consider the aforementioned and other subtleties.

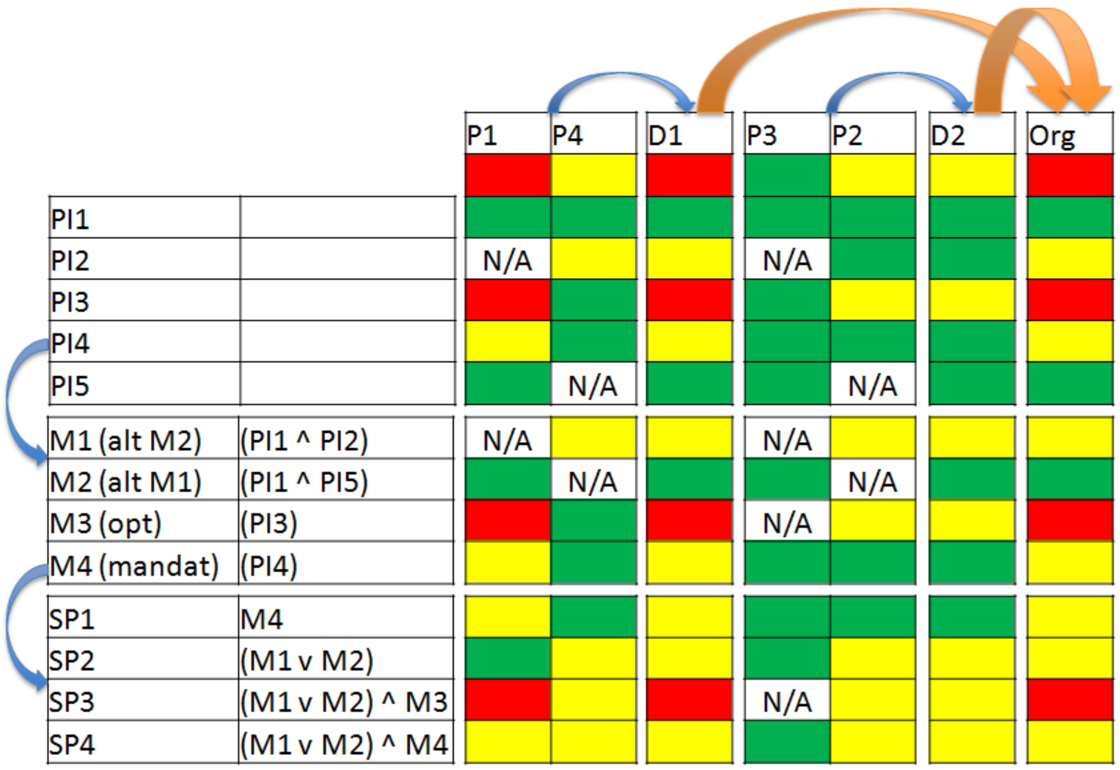
The example represented in Figure 4.10 demonstrates the aggregation of results from a PI into a method and then into a goal or practice. The organisation has several methods that are part of its processes. Some methods are mandatory, others are optional and in some cases there is a pool of alternative methods and the team can choose one of them. The methods to use can also be imposed by the lifecycle in case of a development process or the methodology used for project management (for example use Scrum or TSP). When a project begins it is necessary to choose the methods that are going to be used. In the figure methods 1 and 2, M1 and M2, are alternative methods to do the same activity, M3 is an optional method and M4 is mandatory.

Taking the example of project P1, they chose the estimation method M2 and have to monitor PI1 and PI5. At the moment at which the project is analysed both indicators are green. The team opted to use an optional method M3, monitored with PI3 and had to use M4, because it is a mandatory method, and consequently monitored PI4. Since PI3 is red the project is red and so is M3. For that project M2 is green because both PI1 and PI5 are green and M4 is yellow because PI4 is also yellow.

When analysing CMMI specific practices we can see that SP1 uses M4, SP2 is mapped with M1 or M2, SP3 with M1 or M2 and M3, and SP4 with M1 or M2 and M4. For project P1 SP1 is yellow, because M4 is yellow, SP2 is green because M2 is also green, SP3 is red, because the team decided to use the optional M3 and it is red and SP4 is yellow because M4 is yellow as well. The project is red in terms of PI, methods and SP.

If a department has several projects, it may evaluate its state by aggregating the results of its projects. In the case of department D1, P1 is red which implies that the department is red too. The department can also analyse its results in each PI by aggregating the PI results in each one of its projects. In the case of D2, PI3 is yellow because in one of the projects the indicator was yellow and the other did not have a red. A similar analysis can be done to evaluate how the department is performing each method and each SP.

To evaluate the organisation, the analyst aggregates the results of its departments, in this example D1 is red so the result of the organisation is red. The organisation can also evaluate each PI, method and CMMI goals or practices by aggregating the results of each department. For example, SP1 is yellow because D1 is yellow and D2 is green. To evaluate the organisation, the analyst



		P1	P4	D1	P3	P2	D2	Org
		🟥	🟨	🟥	🟩	🟨	🟨	🟥
PI1		🟩	🟩	🟩	🟩	🟩	🟩	🟩
PI2		N/A	🟨	🟨	N/A	🟩	🟩	🟨
PI3		🟥	🟩	🟥	🟩	🟨	🟨	🟥
PI4		🟨	🟩	🟨	🟩	🟩	🟩	🟨
PI5		🟩	N/A	🟩	🟩	N/A	🟩	🟩
M1 (alt M2)	(PI1 ^ PI2)	N/A	🟨	🟨	N/A	🟨	🟨	🟨
M2 (alt M1)	(PI1 ^ PI5)	🟩	N/A	🟩	🟩	N/A	🟩	🟩
M3 (opt)	(PI3)	🟥	🟩	🟥	N/A	🟨	🟨	🟥
M4 (mandat)	(PI4)	🟨	🟩	🟨	🟩	🟩	🟩	🟨
SP1	M4	🟨	🟩	🟨	🟩	🟩	🟩	🟨
SP2	(M1 v M2)	🟩	🟨	🟨	🟩	🟨	🟨	🟨
SP3	(M1 v M2) ^ M3	🟥	🟨	🟥	N/A	🟨	🟨	🟥
SP4	(M1 v M2) ^ M4	🟨	🟨	🟨	🟩	🟨	🟨	🟨

Figure 4.10: Aggregation of evaluation in the source perspective and target perspective (Lopes Margarido et al., 2011b).

Legend: PI - Performance Indicator, Org - organisation, D1 - department 1, P1 - project 1, alt - alternative, opt - optional, mandat - mandatory, ^ - AND, v - OR.

aggregates the results of its departments, in this example D1 is red so the result of the organisation is red. The organisation can also evaluate each PI, method and CMMI goals or practices by aggregating the results of each department. For example, SP1 is yellow because D1 is yellow and D2 is green.

The evaluation is based on the results obtained to achieve a goal, compared with the thresholds established by the organisation. For example, if a department has a goal that is common to one of the organisation, and one of its PIs is red, the goal is not achieved. Consequently, the evaluation of such goal will reflect the worst performance, and that will also be reflected on the evaluation of the organisation. The metamodel of Repository and Evaluation along with the Procedures of Setup, Tailoring and Evaluation, contributes to answer research question **RQ4** - *How can we evaluate the quality of implementation of the CMMI practices, ensuring that organisations fully attain their benefits and perform as expected?*

4.4.2 CMMI Implementation: Problems and Recommendations

The module **CMMI Implementation** provides a checklist of problems that may occur when implementing CMMI and a set of recommendations (R) on how to implement the model based on the problems (P), which we numbered. The implementation checklist is based on the literature review that we discussed in 3.1.3 Problems in Process Improvements, Metrics Programs and CMMI, the review of the SEI technical reports (TR) of a survey to organisation aiming to achieve HML we presented in 3.2 Survey on MA Performance in HML Organisations, further analysis that we did of the data of that survey (SDA) and also the case studies that we conducted in three high maturity organisations (CI, CII and CIII). This research is detailed on 5.1 CMMI HML Implementation. We did an analysis of the reports on two SEI surveys (discussed in 3.2 Survey on MA Performance in HML Organisations) and their data (detailed in 5.1.1 Further analysis of the HML Survey Data) to find recommendations for process performance that were related to organisations achieving high maturity.

MA plays an important role in HML, including in the definition and use of PPM and PPB. If the measurement protocols, how to collect and analyse the data, and the data are not consistently collected and meaningful, PPM and PPB will be useless. We complemented the statistical analysis done by Goldenson et al. (2008); McCurley and Goldenson (2010), who analysed the relations between factors that contribute to see value in the PPM implemented, in the SDA that was focused on the factors that are related with organisations achieving the desired CMMI goal.

The literature review showed us that many of the problems found in the high maturity levels of CMMI are actually based in ML2. Understanding the statistical needs to achieve ML 4 is not reduced to having knowledge of statistics and modelling, but requires knowing how to use them in the business area without falling into the traps of blind interpretations, or over-relying on models and baselines, without critically analysing them in the context of the process execution, e.g. context events occurring when a project was using the process. Next we detail the problems and challenges organisations face when implementing CMMI and recommendations to avoid them. In some cases the same recommendation should be followed to avoid more than one problem; we detail the recommendation below the problem it applies to.

Entry Conditions

P1. Underestimate time to implement HML

The time to implement HML can be long, even though there are cases of organisations that can implement HML in shorter periods, for example move from ML 3 to 5 in six months and twelve months (Fulton, 2002). However, to be able to become more mature at such a pace, requires that the organisation already has a good base, and even some high maturity practices in place, and merely has to complement these and fill a few gaps; or, as described by Fulton, senior management and executives are members of the SEPG, and have responsibilities that are part of their job, the implementation has a clear connection to the business case, and have mature historical data available. Organisations tended to underestimate the time needed to implement CMM (Herbsleb and Goldenson, 1996). As seen in Figure 3.3, CMMI implementation requires time: to define processes, implement them, involve the right people, train everyone and refine the practices before they become stable. CMMI just provides guidelines, so it is still necessary to plan the time required to define the processes themselves. Moreover, when considering high maturity there must be cycles to complete data collection and analysis to refine the models, particularly if they are being implemented for the first time. In the TR most of the organisations indicated it took them a long time to accumulate historical data, which indicates they underestimated at least the time to build PPM and establish PPB. It is important to plan all this work.

R1. Plan time for all necessary process improvement activities

When planning a move towards HML time needed must be respected to: have mature levels and institutionalised practices; understand and analyse the needs for HML; find correlations between variables; reach stable metrics, processes, tools and work habits; select meaningful performance indicators and gather enough stable data points to have statistically meaningful historical data. Some of the TR organisations remedied the time it takes to gather enough historical data by doing real time sampling of processes. Organisations need to carefully plan business and process improvement objectives, temporal horizon and resources (time, internal and external human resources, tools, training, etc.).

P2. Introduction of HML forgetting ML 2 and 3

CMMI practices of a higher level are built over the prior levels and depend on their stability and "maturity". Organisations may implement ML3 without considering HML such that they do not have the right mindset and a solid basis when implementing LMLs (Low Maturity Levels) before moving to HMLs (Leeson, 2009).

R2. Have mature and stable levels 2 and 3

HMLs only work with a stable base (Leeson, 2009), CMMI builds maturity/capability that is base from one level to the next. Hence, introducing HML practices can only occur after ML 2 and 3 are mature and institutionalised, meaning that their practices, metrics and standard processes must be well defined as requirements for the practices of levels 4 and 5.

P3. Understand the statistical/quantitative nature of level 4

Some organisations do not understand the statistical nature of ML4 (Hollenbach and Smith, 2002; Takara et al., 2007). Level 4 is built using statistics but is reduced to them. HMLs require understanding statistics, modelling and quantitative management. The change of mentality from ML3 towards statistical and quantitative thinking takes time and is a necessary condition to implement HML. In TR some organisations revealed the following as obstacles: management and other relevant stakeholders not involved in models decisions, PPM modellers did not have access to statisticians, insufficient alignment of MA with business and technical goals and even more emphasis on statistics rather than domain knowledge, leading to ineffective models.

R3. Involve statisticians with experience in software and CMMI

Guarantee the involvement of an expert in quantitative methods (e.g., statistician), preferably with experience in software and if possible also in CMMI, who can help to better understand processes behaviour and correlations between variables, along with providing adequate statistical tools to different contexts. From the TR results, to build valuable models and achieve HML, substantially use statistical techniques and methods such as regression analysis for prediction, analysis of variance, SPC control charts, designs of experiments. For modelling use simulation or optimisation methods: Monte Carlo simulation, discrete event simulation, Markov or Petri-Net models, probabilistic models, neural networks, optimisation. The models should put emphasis and meet purposes consistent with the "healthy ingredients" and diversified to predict product quality and process performance.

R4. Introduce Six Sigma

Introducing a Six Sigma initiative in the organisation eases the introduction of the statistical knowledge necessary to the organisation workers and provides the necessary tools to implement HML (Hefner, 2009).

R5. Top down and bottom up goals review

There must be a top down and bottom up revision of the organisation's processes, improvements/innovations, goals and quantitative goals.

Process Definition and Implementation

P4. Processes copied from the CMMI model

As stated before, CMMI provides guidelines, not processes. Using the model as is and considering it the process, neglects many details of how to do the work, hardly reflects organisation reality and culture and may not add value. Even copying processes from other organisations may not give the intended results (Diaz and Sligo, 1997).

R6. Processes reflect organisation's culture

The implementation of the model should reflect the culture of the organisation, and not be a copy imposed on personnel (Diaz and Sligo, 1997). Processes definition should identify current processes (*as is*) and improvements (*to be*) so they reflect an organisation's culture and people good practices (Leeson, 2009).

R7. Involve experts and users of the processes

When defining processes it is important to involve the experts, including those who use the process to do their work such as project, technical and quality managers, developers, and testers (Diaz and Sligo, 1997).

P5. Multicultural environment

Difficulties in having a common organisation culture in multicultural environments, reflecting in processes that "fit all", that everyone recognise and use.

R8. Promote processes sharing and lessons learnt amongst different BU teams

In multicultural organisations and when acquiring new companies imposing processes can result in a loss of knowledge and resistance to change. Different business units should share practices used and lessons learnt. Each business unit would then gradually and naturally adopt the other's practices if they better fulfilled needs. This approach allows creating processes without losing good practices, while benefiting from cultural differences.

P6. Impose processes

People are forced to use the processes, at times without understanding them or without having opportunities to present other ways of work and improve the process if the shared ideas are good. Happens in some management styles and when acquiring new companies, for example.

R8. Promote processes sharing and lessons learnt amongst different BU teams (as defined in P5)

R9. Goals specific to different organisation levels, related to the organisation's business goals

The processes must reflect the organisation's culture that is also why people must be involved in process definition. An additional benefit of this involvement is that it helps people understand, relate with, and embrace the changes. There should be goals specific for different business units, departments and projects, which must be related to the organisation business goals. The transition to HML is facilitated when there is clearly tied to the business case (Fulton, 2002).

R10. Monitor at different report levels

Having the setting in R9 allows having goals monitored at all levels, avoiding the loss of visibility by middle management in each level.

P7. Dissemination problems

There are people who do not know the process. People must be aware of the changes taking place and have all information necessary to do their work once they start using the new processes; hence the importance of coaching, training and providing tool tips and help with the software to use. TR revealed cases of insufficient mentoring of process modellers.

R11. Commitment from the entire organisation

Commitment from the entire organisation is essential, including top management, middle management (Diaz and Sligo, 1997) and the people who are actually doing the work (Leeson, 2009).

R12. Complete and adequate training

Training needs to be adequate for each role and to include not only the *what to do*, *how to do* and *hands on* components but also the *why shall we do it*, *what will we achieve* and *how do we see it*. However, when organisations are large they should consider even more gradual dissemination, spreading practices in a small group of projects and gradually involving new ones, which can be done also profiting from team members' mobility.

R13. Projects and people coaching

To have people commitment it is crucial that they understand the new practices, which can be achieved by coaching projects and people (Humphrey, 2006), guiding and accompanying them.

P8. Lack of institutionalisation

The standard processes are not being used by everyone or by all projects for which the process is applicable. Radice (2000) referred to this problem as lack of institutionalisation, the SEI indicated that there were practices not used organisationwide (Schaeffer, 2004), Hamon and Pinette (2010) found processes that were not followed or the cases reported by Charette et al. (2004) of lack of adherence to them. Besides ensuring the right level of dissemination and people involvement, to have institutionalisation also implies that different contexts are understood and contemplated in the program.

R14. Top management: set goals, plan, monitor and reward

Top management needs to set goals, plan gradual institutionalisation, monitor and reward. That is why R12 is important, it is essential that upper managers understand the processes by receiving adequate training.

R13. Projects and people coaching (as defined in P7)

R15. Mature processes and metrics with practice

Metrics and processes definitions mature when used in practice because it is when problems arise that it becomes more evident how procedures can actually be done. It is necessary to give some time to let processes and metrics mature before producing their final versions.

Metrics Definition

P9. Meaningless metrics

Metrics that are irrelevant for the business, that people do not understand nor use. This problem was analysed by Kitchenham et al. (2006) who found that the metrics were irrelevant for the upper management. The metrics used should be useful, that is to say, provide information that is needed to achieve a goal. Furthermore, one must not fall in the temptation of blindly trusting causality or establish relations between what is not correlated.

P10. Metrics definition (collect and analyse data)

The metrics are not clearly defined, including the procedures on how to consistently collect and analyse them. The quality of metrics definition was one of the concerns of Goulão (2008); Hamon and Pinette (2010) and Barcellos (2009). Define metrics unambiguously, ensure that they are measuring exactly what is needed and know how to interpret them.

R16. Measurement reflecting goals

To establish business objectives and identify the indicators of the processes performance, organisations can use methods such as the goal-driven measurement (Park et al., 1996).

R18. Appropriate metrics

The metrics to use need to be appropriate in the context, so if new contexts arise the process needs to be updated. Understanding metrics is completed only when projects are using them as the final processes define. It is utterly necessary to train the entire organisation without undervaluing the effort in such tasks.

R17. Measurement and analysis protocols

Measures need to be defined with a set of repeatable rules for collecting and unambiguously understanding data and what they represent (Florac et al., 2000), if different people use them differently, then their definition is inadequate. The level of detail of metrics needs to be completely defined and understood. In TR the organisations achieving the HML goal had good documentation relative to process performance and quality measurement results.

R19. Measurement context

Additionally, it is also important to consider the different types of projects' context, including the technology used. For example, in some technologies there are more KLOC, the time to execute unit tests is negligible, etc. Another example is project type: outsourced, maintenance and development projects, for instance, will have different measurement and control needs. Those factors affect the metrics definition.

R20. Normalise variables

Define basic software processes about which data should be collected, then concatenate and decompose data in different ways to provide adequate information at project and organisation levels (Kitchenham et al., 2006).

P11. Uncorrelated metrics

The metrics to build the processes performance models or baselines are not related to the process, not correlated, are collinear or there is statistical correlation between variables that are unrelated. This problem was also identified by Kitchenham et al. (2006). Hamon and Pinette (2010) also indicated that at beginning there may be too many indicators. The first data collection may not provide the correlations the organisation is looking for, so it has to refine and perform new data collection cycles.

P12. Metrics categorisation

In the first cycles of data collection, at times the baselines are not stable enough. Also, unless there is already a considerable amount of historical data, it is not possible to distinguish between different categories of data (for different markets, team experience, team sizes and project sizes). The most common obstacle reported in the TR was not having enough contextual data for data categorisation, some organisations also revealed that their metrics were not consistent for data aggregation. Depending on the context some metrics may not be adequate to measure everything, for example, the defined size metrics need to be related with the nature of the work being measured. Furthermore, data collection needs to be consistently done.

P13. Baselines not applicable to all projects

Having unstable PPBs not specific to the different contexts. Time to collect data insufficient to gather information of different contexts and verify if:

- New metrics are needed;

- There are differences in performance, and in which context;
- In certain circumstances, the procedure to collect the data should be different.

R24. Aggregate normalised data for global view

If necessary, data should be normalised to make them visible to top management.

R21. Gradual data collection

It is preferable to begin with a sub-process executed often and with a small number of variables so results come faster. When the process is stable, then extend to other processes and more complex ones (Florac et al., 2000).

R23. Categorise data

Metrics databases take time to become stable and allow the construction of relevant PPM and PPB. The data to be categorised. (Florac et al., 2000) refer to this process as "separating or stratifying data that belong to the same cause system".

R22. Evolve baselines in time

Nonetheless, to have adequate categorisation it is necessary that the different projects fully cycle to completion. Either the organisation has a significant number of concurrent projects with small lifecycles or it begins to work with first limited baselines that evolve with time. Stable databases to develop relevant PPM and PPB take time. In TR it was shown a relation between achieving the desired HML and regularly using PPM in status and milestones reviews.

Pilot projects are useful for stabilising processes, procedures and tools. The way people use tools may change the way metrics should be collected. Only after those projects are over, and the practices are clearly defined, will the organisation be ready for training, the processes/procedures and tools be fully and correctly documented and people be able to learn and apply the practices. Changes may then be deployed so that processes become institutionalised.

Metrics Usage

P14. Abusive elimination of outliers

Outliers eliminated without understanding whether they were special causes of variation or not; for example discarding relevant information to the execution of the project. Some data points are recurrent in all projects, for example, therefore, they should be considered as part of the process behaviour.

R25. Recognise special causes of variation

Certain outliers can be removed from databases but it is necessary to pay attention to those not immediately understood. They can indicate that a process is having a new behaviour (better or worse performance), be a common situation or indicate the existence of a different process, with a different behaviour and therefore originate new sub-processes (Florac et al., 2000). Some outliers are just indicators that the process improved its performance (Spirula Member, 2010, personal communication).

R26. Quarantine outliers that are not understood

One way of avoiding the error of abusively eliminating such outliers is to monitor the process without the outlier in parallel to the process with the outlier, then decide the most adequate action:

- Perform CAR;
- Eliminate the outlier;
- Establish a new baseline because process performance improved;
- Create new sub-processes, in case of having sub-processes.

Florac et al. (2000) give an example of how to do it.

P15. Data not being collected in all projects

Not all projects gather data, a problem which occurred in TR. Different projects cycles require tools adaptation and in some cases specific metrics, to be measurable.

R27. Use adequate base measures

Define the base measures that are appropriate to the different work.

R14. Top management: set goals, plan, monitor and reward (as defined in P8)

P16. Effort estimates

Not having different support baselines/tools for different effort estimation methods. For example, having expert judgement without the support of previous knowledge of work-product size and task duration, simply because there are only PPB adequate for code development projects.

R28. Use expert judgement to estimate when needed

Regarding effort estimation, expert judgement is more appropriate in certain circumstances, in particular when there is absolutely no previous knowledge of the project (Grimstad and Jorgensen, 2006).

R29. Use related historical data

Effort estimation does not necessarily need to be based on KLOC to be based on historical data; it can be based on other size metrics, phase duration or the time spent on task.

R30. Build historical database by planning iteratively

When no data are available at all do iterative planning, so that when data from a previous cycle are available they can be used to plan the following.

P17. People behaviour

One of the challenges of process improvements, and CMMI is no exception, is overcoming resistance to change (Diaz and Sligo, 1997). In TR the respondents revealed that one of their of their issues was people resisted to collect new or additional data after achieving ML3. This problem can happen due to failing to show people the value of practices that should be applied on their projects or work, and consequently having people not using them or considering those practices are not applicable to their projects. This behaviour may also result in careless data gathering, compromising their accuracy.

R12. Complete and adequate training (as defined in P7)

R13. Projects and people coaching (as defined in P7)

R31. Never use personal data to evaluate people

To have useful, reliable data, personal data shall not be used for evaluation purposes.

R32. Data quality and integrity checks

TR showed the importance of doing quality and data integrity checks that are also useful to prevent that the data gets compromised due to any issues that may occur while they are being collected.

Tools Setup

P18. Tools setup and requirements stability

Tools that are incomplete, present defects or are not adequate to be used in practice. Tools require time to be stable, new defects can be found only when they are already in use and changes to the original requirements may be revealed by usage in work context.

R33. Improve tools with usage

It is important to understand that tools need time to be set up, especially when evolving existent ones.

R34. Let tools and processes stabilise before considering the collected data

The data collected when correcting those tools defects, which have impact on the definition of the metrics and of the process, should not be used to build PPB because the process is not stable. For the same reason, PPM may also need to be recalibrated, for example.

R35. Guarantee collection process precision

It is imperative that data collection is precise, if it was not so previously, people need to change their mentality and display discipline.

P19. Overhead

Having additional work to do with the introduction of the new practices that is not contributing to the outcome of the process and could be automated. When data collection is not fully automated, doing it manually may introduce overhead, besides increasing error caused by human mistakes. The TR includes cases of people considering there is too much time spent reporting data rather than analysing them. Monteiro and Oliveira (2011) also indicated that organisations can define too many metrics that are the not used or analysed.

R36. After PPM and PPB stabilisation only collect necessary data

Once processes and their PPM and PPB are finally stable, and the needs are completely understood, only collect the data that is necessary.

R37. Use automated imperceptible data collection systems

Manual data collection is time-consuming and error prone (Hamon and Pinette, 2010), so it should be automated. To avoid overhead in the collection process, the information system needs to have limited human intervention, e.g., reporting effort and measuring code. Effort spent on different software applications for doing the tasks may be measured and part of the effort automatically labelled; the person only verifies and corrects eventual errors by the end of a block of tasks. This avoids forgetting to report effort or constantly interrupting tasks to manually report. The information system should be composed of automatic storage tools connected to the development environment (Johnson et al., 2005).

We compile all problems (P1 to P19) and recommendations (R1 to R37) in a checklist (see Table 4.1) to be used by organisations when implementing CMMI. For some of the problems, we also indicate the PA, SP, generic goal (GG), or generic practice (GP), which are possibly affected. The checklist provides guidance as to the sequence of what should be done to implement CMMI, gives organisations focus on the model as a whole, not only a single target level to be achieved, and includes the problems that organisations should be aware of in order to avoid them.

The problems we found can go from the implementation of CMMI, to the usage and interpretation of the processes' outputs; we also compiled corresponding recommendations to help avoid them. Therefore, we can answer the research questions **RQ1** - *Why do some organisations not achieve the expected benefits when implementing CMMI?* and **RQ3** - *What additional recommendations can we provide to organisations to help them avoid problems when implementing CMMI?* Furthermore, we recommend that appraisers look for evidences of these problems and recommendations when evaluating organisations, in order to sustain the achievement, or not, of a given maturity level.

When using EQualPI to support the implementation of CMMI, the organisation can load their processes and methods following the setup and tailoring steps, described in (4.4.1 EQualPI Setup, Tailoring and Evaluation). After setting up the Framework, the organisation will have the methods aligned with practices and performance indicators. The next step is to load the data. The processes may be unstable but the results shown by the Framework will reflect the processes and data in place in the organisation at a given time. Therefore, as processes are changed so is the Framework. The organisation will be able to see the evolution of their CMMI implementation over time.

Table 4.1: CMMI Implementation Checklist: list of activities to follow in order to avoid common problems.

	Problem	**Orgs.**	**PA/GG**	**Recommendations**
Entry Conditions	*P1. Underestimate time to implement* (CMM Herbsleb) HML: it takes long to accumulate meaningful historical data.	CI,CIII ,TR		R1: Plan considering activities such as maturing levels, analysing and understanding HML, maturing PPB and PPM, collecting data repeatedly until meaningful performance indicators can be systematically obtained. Do real time sampling process to shorten time to gather enough historical data (TR).
	P2. Introduce HML forgetting ML 2 and 3 (Leeson)	CIII		R2: Before moving to HML guarantee that ML 2 and 3 are mature and institutionalised.
	P3. Understand the statistical/quantitative nature of level 4 (Holenbach, Takara): Underestimate time to change mentality from ML 3 to quantitative thinking, and to implement ML 5. Insufficient involvement of management and stakholderss in models decisions. Ineffective models.	CI,CIII ,TR	MA OPP QPM	R3: Involve a statistician with experience in software and preferably also in CMMI. Substantially use statistical techniques and simulation or optimisation methods (TR). Build models that put emphasis on the "healthy ingredients". R4: Introduce Six Sigma initiative (Hefner). R5: Review goals and quantitative goals top down and bottom up when implementing CMMI.
Process Definition & Implementation	*P4. Processes copied from the CMMI Model*	CII		R6: Processes shall reflect the culture of the organisation, not be a copy of the model imposed to the personnel (Leeson, Diaz). R7: Involve experts and process users in the definition of processes (Diaz).
	P5. Multicultural environment: people dealing differently with change.	CII		R8: Interaction between business units to share processes and lessons learnt to design processes together.
	P6. Impose processes on acquired organisations with the loss of good practices.	CII		R8,R9: Have goals specific to different business units, departments and projects, related to the organisation goals, tied to the business case (Fulton). R10: Have indicators to monitor them at different report levels.

Continued on next page

Table 4.1 – *Continued from previous page*

	Problem	Orgs.	PA/GG	Recommendations
	P7. Dissemination problems: difficulties in applying new practices, in particular in understanding how to collect, analyse and interpret metrics. Managers using PPM/PPB do not understand results and PPM modellers lacking mentoring/coaching and access to a statistician.	CI,TR	GP2.5 GP2.6	R11: Have commitment from the entire organisation: involve top management, middle management and the people who are actually doing the work (Diaz, Leeson). Have a sponsor (Leeson). R12: Training shall include what to do, how to do, hands on, benefits and how can benefits be seen. Have different levels of training. Specialised training for sponsors and top management, process group and all roles that are affected by changes. Train top management on: sponsorship; goal setting; monitoring and rewarding (at different goals levels); on the process (understand it). R13: Coaching of projects and people (guiding and accompanying) and monitoring (from top management) (Humphrey).
	P8. Lack of institutionalisation(Radice, Schaeffer, Charette, Hamon): not all projects used the new practices. Lack of adherence to processes or unused processes.	CI,CII	GG 2 GP2.5	R14: Top management: set goals (when, who, what); include goals for gradual institutionalisation, monitor and reward. R13, R15: Metrics and processes definitions mature when used in practice, need time to define final versions.
Metrics Definition	*P9. Meaningless Metrics*(Kitchenham): misinterpretation of metrics due to lack of context information; useless indicators and at times not aligned with business and technological goals.	CI,TR	MA SP1.4 MA SP2.2	R16: Use goal-driven measurement (Park), or equivalent, to establish quantitative goals. R17: Measures defined with a set of repeatable rules for collecting and unambiguously understand the data and what they represent (Florac). R18: Use metrics appropriate to the context (e.g. different size measures according with the work product) and as the final processes define. R19: Identify different context that need to be associated with the metrics in order to correctly interpret them. R20: Do variables normalisation to ensure that metrics are usable in the entire organisation and at different organisation levels (Kitchenham).
	P10. Metrics definition (collect and analyse data)(Goulão, Hamon, Barcellos): not adequate to all contexts, vague, allowing errors in collected data due to different interpretations, inconsistent measures for aggregation across the organisation.	CI,CII	MA SP1.3 MA SP1.4 MA SP2.1 MA SP2.2	

Continued on next page

Table 4.1 – *Continued from previous page*

	Problem	**Orgs.**	**PA/GG**	**Recommendations**
	P11. Uncorrelated Metrics: first data collected were uncorrelated (Kitchenham). Too many indicators at the beginning (Hamon).	CIII	OPP	R21: Conduct all necessary data collection cycles to find correlated metrics (Florac). R22: Give time for the metrics databases to become stable and allow the construction of relevant PPM and PPB; and for different projects' full cycles to be completed. R23: Categorise data (Florac). R24: Aggregate normalised data only for global view.
	P12. Metrics Categorisation: not all contexts data available. Unstable baselines without different categories. Not enough contextual information for data aggregation.	CI,TR	OPP	
	P13. Baselines not applicable to all projects	CI,CII	OPP SP1.3 OPP SP1.4 QPM SP2.2	
Metrics Usage	P*14. Abusive elimination of outliers*: exceptional causes of variation occurring once per project or new baseline being established.	CI	MA SP1.4 MA SP2.2	R25: Recognise data points that are not outliers but are unique and recurrent (Florac, Spirula). R26: Quarantine outliers which cause is not immediately identified (Florac).
	P15. Data not being collected in all projects: not collecting data from projects with a data structure different from the standard. Not using all derived metrics because of lack of definition of base measures adequate to context.	CI,CII ,TR	MA SP1.3 MA SP2.3	R27: Base measures should be defined for different work and, when needed, normalised to allow calculating derived measures. R14
	P16. Effort estimates: without using historical data of effort or size.	CII	PP SP1.2 PP SP1.4	R28: Expert judgment is more adequate in certain circumstances (Grimstad). R29: Use any related historical data: size, phase duration, time spent on task. R30: Do iterative planning and use real time sampling of processes when there is no previous data available (TR).

Continued on next page

Table 4.1 – *Continued from previous page*

	Problem	Orgs.	PA/GG	Recommendations
	P17. People behaviour(Diaz): inaccurate personal data reports. Resistance to collect new/additional data after ML3.	CI,CII ,TR		R12, R13, R31: Never use personal data to evaluate people. R32: Perform data quality and integrity checks (TR).
Tools Setup	*P18. Tools setup and requirements stability*: problems in tools after deployment. New needs still being identified, new tools still being developed. Using the tools in practice and in different projects contexts allowed to identify undetected problems and necessary improvements.	CI	OPM SP2.2 OPM SP2.3	R33: Tools are improved when used in practice, save time for their setup. R34: When correcting tools that have impact in the metrics definition and the process, do not use the collected data to build PPB. R35: Guarantee that data collection is precise (discipline people and change their mentality).
	P19. Overhead: in tools usage (data collection not completely automatic) and too many metrics not being analysed (Monteiro).	CI		R36: Once PPM and PPB are stable only collect data that is needed. R37: Use automatic and unperceived data collection systems (Hamon, Johnson), with limited human intervention (start/stop and confirm).

We identified authors who found the same problems and gave the recommendations by the first author's surname. For the complete reference please check the description of the problems and recommendations given. Orgs. refers to the organisations where the listed problems were found, either in our case studies (CI, CII and CIII) or the SEI survey (TR).

4.4.3 Process Improvements

Doing a process improvement involves steps that are common, regardless what the organisation intends to change, but also has particularities. There are several methodologies that can be used such as PDSA, Quality Improvement Process, Six Sigma's DMAIC or IDEAL, and it is not our intention to provide a new one. Explaining how process improvements shall be carried out, is in any case outside the scope of this thesis. Therefore we will stick to the steps we followed in a process improvement experiment we did (detailed in 5.2 Requirements Process Improvement), and highlight important elements to consider while carrying out the process improvement, particularly to complement improvement steps with the scientific method (see Figure 4.11). Our improvement was piloted with undergraduates and graduate students, and was later implemented in an HML organisation that is currently using it.

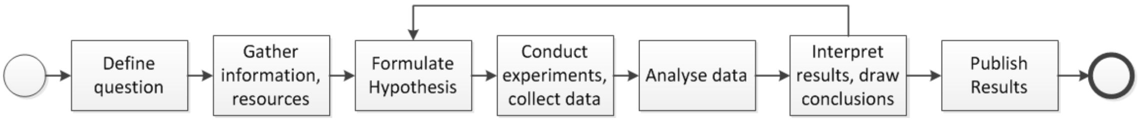

Define question
Gather information, resources
Formulate Hypothesis
Conduct experiments, collect data
Analyse data
Interpret results, draw conclusions
Publish Results

Figure 4.11: Representation of the scientific method (Goulão, 2008).

Process improvement is a continuous activity of organisations, so there would always be future improvements to restart the Process Improvements procedure. We represent its steps in the diagram in Figure 4.12.

Step 1 - Identify a need and characterise the current process

The first step of process improvements is the identification of a need. For example: an organisation may want to achieve a business goal that cannot be attained in the current organisation setting, or the organisation may be facing a problem that needs to be solved as it is affecting business. It is similar to what Juran and Godfrey (1998) called "Awareness: proof of the need", in DMAIC is mappable to the Define phase (Hahn et al., 1999), while in the IDEAL model this would correspond to the Initiating phase, more specifically "Set Context" (Gremba and Myers, 1997). In PDSA (Moen and Norman, 2006), the Plan would be comprised of the first 4 steps we defined.

- In our experiment the **need** was to reduce the number of defects from the requirements phase, only detected in posterior phases of the software development cycle.

When such a need exists it is necessary to understand the reason why the need is not fulfilled in the current setting, the root cause of the problem, or which processes can be changed to achieve the goal.

- Our target process area was Requirements Management more specifically the process **Review Requirements**. In the case of our process improvement, to detect defects more effectively in requirement reviews we assembled a defect classification taxonomy, specific for requirements. We

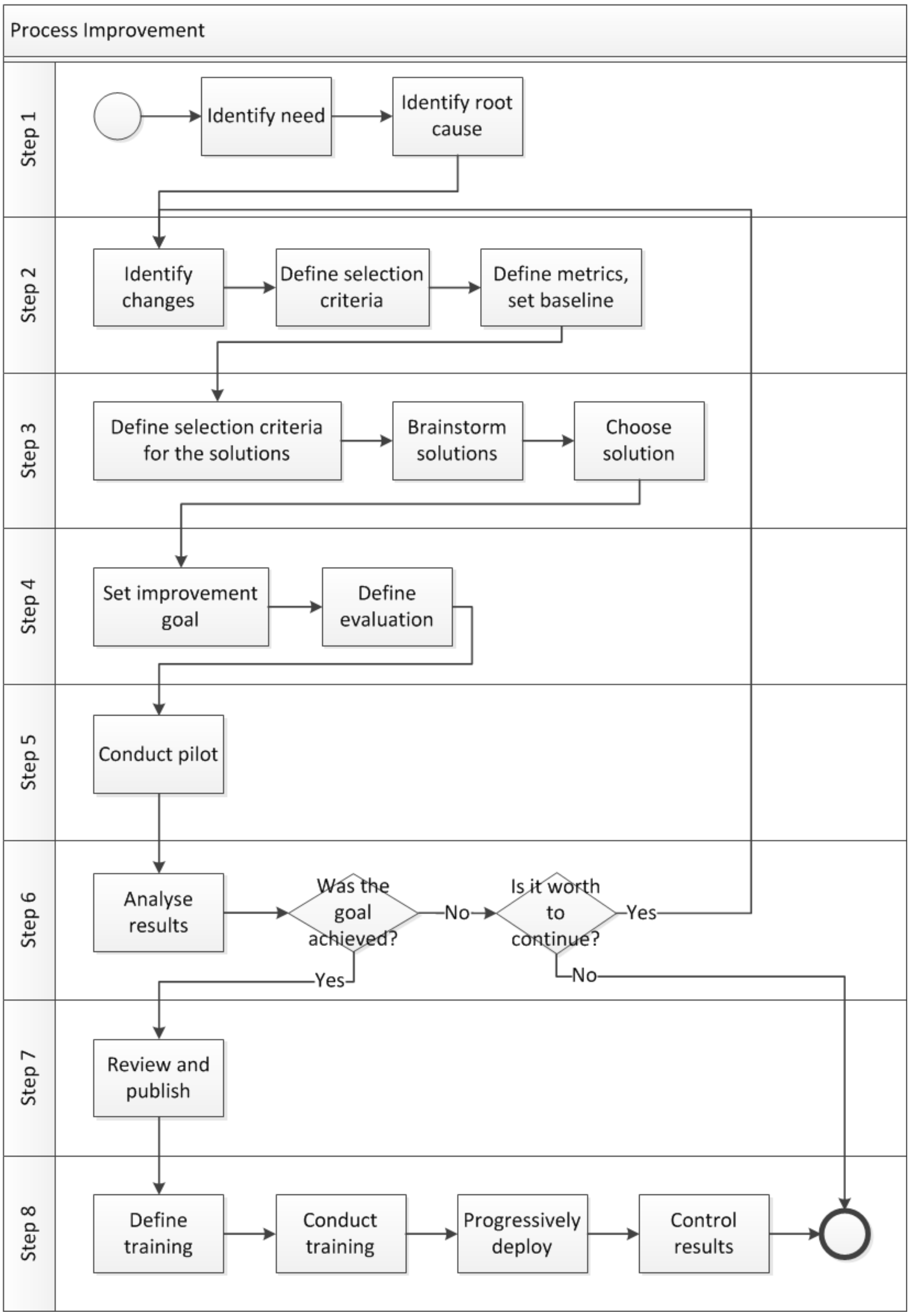


Figure 4.12: Process improvement steps.

could have used the IEEE Std 1044-1993 standard to classify the type of defect, using the classifiers of **Documentation problem** and **Document Quality problem** but the complete list includes fifteen classifiers, too many when compared with the recommended limit of nine, as we followed the Freimut et al. (2005) quality criteria in Step 3. In the most recent version of the standard IEEE

Std 1044-2009, the attribute Mode appears to be the one with values more adequate for requirements: **Wrong**, **Missing** and **Extra**. The values of the attribute Type, seem to be more suited for specifications of design and architecture, and code defects, but the following could be considered to classify requirements defects: **Interface**, **Logic**, **Description** and even **Standard**.

Step 2 - Identify and define improvement

It is necessary to determine the changes to implement, which methods will be used and which metrics can be used to monitor changes. In the case of having a problem to solve whose origin is unknown, one must select an adequate set of metrics to monitor and analyse them in order to determine the root cause of the problem and help define a solution for it, which in DMAIC would correspond to the Measure and Analyse phases (Hahn et al., 1999). In any process improvement it is necessary to have selection criteria to determine the methods to use.

Organisations need to be able to measure improvements otherwise it is not possible to determine if they were beneficial or worsened the situation. For that purpose it is necessary to determine the performance indicators that need to be monitored and create a baseline of those indicators. When EQualPI is used to monitor process improvements that task is eased because it is even possible to detect the effect of the improvement in other processes by monitoring other indicators. The baseline provides the state of the current process, *as is*, including the indicators values in that state.

- In the case of piloting our improvement in an organisation, it would be necessary to understand the number of defects that were detected in posterior development phases that were due to requirements defects. Therefore, the metrics to monitor would be the defects per phase originated in the requirements phase.

This step would be part of the "Diagnostic Journey" (Juran and Godfrey, 1998), and in DMAIC would correspond to the Measure and Analyse phases (Hahn et al., 1999). In IDEAL this step would be part of the Diagnosing phase when characterising the current solution and developing recommendations (Gremba and Myers, 1997).

Step 3 - Determine selection criteria and select improvement methods accordingly

With the improvement identified it is necessary to define the selection criteria for the solution to pilot and brainstorm the possible solutions involving experts and people who will use the process.

-In our case we had to assemble a classifiers list. We reviewed the literature to find what defects classifications were used and which ones were specific/more adequate to classify requirements defects. Another aspect we had to find was what recommendations there were on how to correctly define a classification list.

The solution is selected based on the selection criteria.

- Freimut et al. (2005) indicated the quality properties of a good classification scheme, that we used as a reference while assembling the classification of requirements defects.

This step is similar to the formulation of theories that must be tested in order to select the remedy to apply (Juran and Godfrey, 1998). In the IDEAL model it is part of the Establishing phase, by setting priorities and developing the approach and planning the actions, but also would be creating the solution, which is part of the Acting phase (Gremba and Myers, 1997). In DMAIC this, and from step five to six, would be part of the Improve phase (Hahn et al., 1999).

Step 4 - Set improvement goals and how to validate them

Organisations have to ensure that improvements are not only effective but also efficient, in that they must guarantee a return of investment (ROI). A simulator could be used to predict the ROI of the improvement, where the costs of training people and updating tools would also be considered. This is what Juran and Godfrey (1998) call "the potential return of investment". This step also includes the definition of how the goals will be validated, what the necessary metrics will be and which analysis must be done to ensure the goal is achieved. In the IDEAL model this step is part of the Diagnosing phase, by characterising the desired stage, as well as part of Establishing, as developing the approach and planning the actions would also have to consider what measurements would be needed to validate the improvement (Gremba and Myers, 1997). In DMAIC this step would be part of the Define phase (Hahn et al., 1999).

- If our improvement was to be done in an organisation it would be necessary to analyse the costs of requirements reviews and of fixing defects in development phase. The current costs and goals to achieve would have to be documented. In an organisation setting, to test our process improvement, the final number of defects per phase originated in the requirements phase would have to be reduced to consider the improvement successful. In our setting, we had the goal of ensuring that the classification list was useful to classify requirements defects and classification by different people would be uniform. We designed our experiment to ensure people understood the list, the classifiers and defects were not mixed up, allowing different people to classify them the same way.

- The validation was done by analysing the misclassification (Leek, 2013; Hastie et al., 2009) (equation 4.2), since we expected specific classifiers to be used. From equation 4.3 we derived a similar one (equation 4.4) that we named divergence, to measure the level of concordance of individuals when classifying the same defect.

$$Missclassification = 1 - \hat{}p_{mk(m)} \tag{4.2}$$

Where $\hat{}p_{mk(m)}$ represents the fraction of variables assigned to group m with outcome k.

$$Missclassification = 1 - \frac{TruePositives}{TruePositives + FalsePositives} \tag{4.3}$$

$$Divergence = 1 - \frac{Majority}{Majority + Other} \tag{4.4}$$

- We used the Fleiss' Kappa (Fleiss, 1971) to measure the level of agreement between subjects to classify the same defect and test if that choice was not by chance.

- We wanted to analyse the highest proportion of subjects classifying the same defect unanimously to see if there was a consensus in the classification or not, to understand if the answers could be considered similar or if there were at least two statistically significant proportions (Pestana and Gageiro, 2008). For that purpose we did a Cochran Q test that is binomial, thus we considered that when the subjects chose the most used classifier they answered as the **majority (1)** and when they used any other classifier, they chose **other (0)**.

In any improvement that involves classification, a similar experiment design can be used.

Step 5 - Pilot the process improvement

To pilot improvements it is necessary to select projects for doing the improvement, include a control group that follows the current practices and projects using the new practices. The criteria to select pilots can vary, but here are some examples:

- Projects with good effort and cost margin, to ensure the team has time to follow new practices without compromising the project's successful completion;
- Projects of short duration, to get results faster, in case the improvement does not require a given project size.

The teams need to receive training in the methods that are going to be applied.

- In the case of our experiment we gave the subjects instructions on how to participate in the experiment and included a table with the definition of each defect type and an example of how to use it.

This step is part of the remedial journey (Juran and Godfrey, 1998), and in the case of DMAIC would correspond to the Implement phase (Hahn et al., 1999), while in IDEAL would be part of Acting, namely pilot testing the solution (Gremba and Myers, 1997). This step would be part of the PDSA Do phase (Moen and Norman, 2006).

Step 6 - Analyse pilot results

Analyse the results of the pilot to verify if the improvement actually occurs, and analyse the impact on the indicators monitored to evaluate whether the improvement goal was achieved or not. In case of needing a new pilot, repeat step 5; if it is necessary to find another solution, repeat steps 2, 3, 4 and 5, as needed. The fact that we want to improve and refine the solution if we do not have yet a suitable solution that makes the goal achievable, makes us map this step with the DMAIC phase Improve (Hahn et al., 1999) and the PDSA Study phase (Moen and Norman, 2006). We consider that Juran and Godfrey (1998) improvement's process retrospective analysis and lessons learnt would be applicable at this stage of the process improvement. In IDEAL this step would be part of the Learning phase, when analysing; and validating of the Acting phase, when refining the solution (Gremba and Myers, 1997).

Step 7 - Prepare final version

Publish the final version of the improvement and update tools, processes, templates. Also ensure that they are adequate for people to use them in practice. It may be necessary to test them in a pilot. In the IDEAL model this step could be considered as part of Acting, implement the solution (Gremba and Myers, 1997). Regarding PDSA this would be part of the Act phase, deciding to embrace the change or abandon it, and if went forward, the next step would also be included in this phase (Moen and Norman, 2006). In Juran and Godfrey (1998) process improvement, this step would include establishing controls to hold the gains.

- In our case this would be to update existing tools, including tool tips to help people remember definitions, including definitions and examples in help. Then, test the tools in pilot projects, to test them and use in practice in order to refine as needed.

Step 8 - Progressively deploy and control

Define the training process, how it will be conducted, the plan and what the training materials are. Conduct the training involving all key users. Gradually deploy the improvement in other projects/to other subjects. Control the improvement variables to ensure there are no deviations from the goal. Also, refine the processes, materials and tools as they are used in practice. Do the progressive deployment as indicated in 4.5 Manage Configurations Module, while controlling the indicators.

In DMAIC this step corresponds to Control (Hahn et al., 1999), in IDEAL it would be part of Acting, implement the solution and the Learning phase, analyse and validate (Gremba and Myers, 1997), while in Juran and Godfrey (1998) process improvement would be the institutionalisation of the process improvement.

With the Process Improvements procedure we complement the answer to research questions **R3** - *What additional recommendations can we provide to organisations to help them avoid problems when implementing CMMI?* and **R4** - *How can we evaluate the quality of implementation of the CMMI practices, ensuring that organisations fully attain their benefits and perform as expected?*

4.5 Manage Configurations Module

To have a CMMI implementation and analyse its performance through time it is necessary to manage the configurations. The module **Manage Configurations** in the business layer is where the configurations of the processes are stored, to be used when necessary. Those include the following implementations:

- **Current** - the implementation that is loaded since the last setup of the Framework;

- **Previous** - configurations of previous implementations are saved in the database. They are loaded when the user intends to see the history of evaluations, as an evaluation is based on the organisation data and the configuration of the Framework;
- **Pilot** - configurations of process improvements in pilot projects. It may be possible that they never leave the pilot state if the organisation realises that they do not benefit the organisation;
- **Deployment** - configurations that result in successful pilots are gradually being deployed in the organisation. The deployment configuration co-exists for a period of time with the current implementation, and when a selected percentage of projects is already following it with the desired performance results, the deployment implementation replaces the current implementation that is saved in the history.
- **Updated** - used to establish the new baseline once the deployment of new configurations is completed.

Current and Previous Implementation

The implementation configuration is the one reflecting the current processes configuration that exist in the organisation. To evaluate that implementation the Framework uses all data of the methods that are currently in place. If a pilot is in place or a process improvement is being deployed, the data of those configurations are not used to evaluate the current implementation. Otherwise it would not be possible to compare a pilot or deployment configuration with the current implementation configuration. The previous implementation is also stored.

Pilot

The organisation may start one or more pilots to do a process improvement. The pilots must use the pilot process as it was designed, not the organisation's current implementation nor other pilots' processes. Based on the current configuration, the organisation selects the change to introduce in the process improvement: changing a method, a procedure, or using a different tool. For that pilot, one or more projects will be done and the related data will be collected, eventually new performance indicators will be monitored. The pilots will be executed for a period of time. The data collected to monitor it, will not be loaded when evaluating the current implementation of the organisation, rather they will be loaded only to evaluate the pilot. To verify if the changes improved organisation performance, that is to evaluate the pilot, for a period of time the pilot data are compared with the data of the current configuration in the same period. If the pilot process shows better performance than the organisation current one, the deployment can be done.

Deployment

To prepare the deployment configuration the current configuration is loaded with the change done by the pilot. The organisation begins gradually deploying the changes done and collects the data on the projects that use the changes. In parallel, the organisation keeps evaluating the current implementation to ensure that the performance of the deployed improvement is better than the one used in the current configuration. However, the deployment may be not static. As people use a new tool or functionality and/or a new methodology, the process may need to be adjusted. So, after the change, new data are collected and compared with the current implementation. When the deployed improvement affects a significant part of the organisation (and that is a decision for the organisation to make) and the process and tools are not updated for a given period of time, the improvement is compared with the current implementation. If the performance is considered to be better, and other process areas are not negatively affected, the deployed improvement may become the current implementation. The organisation may keep the improvement and involve the entire organisation, updating the current implementation accordingly. That only happens after training and having tools in place so everyone is ready to use them. Furthermore, if the organisation has a process performance model and baseline, they need time to become stable to be able to compare the new performance. This may be one criterion for the organisation to keep the changes in deployment longer.

Updated Implementation

The new configuration can be loaded from the stable deployment configuration. A new baseline is established. If the data since the final deployment is stable, it may be loaded in the current implementation and the baseline may be set from there. From that point on, the whole organisation operates with the new process and that is the one that is evaluated.

A configuration reflects the implementation of the processes, the methods that implement them, and the performance indicators that are measuring them. When an organisation wants to do a process improvement, defines it and sets the goals for the pilot. The organisation can have one or more pilots in progress, some may not strive, while others will achieve their goals. Once the goals of the pilot are achieved, the final version of the process is prepared and progressively deployed in the organisation, so a new current configuration takes place, and the previous one is stored. At any moment in time, the organisation can see how was the performance in a previous configuration, what is the current performance, and check the performance of a process improvement that is still not ready for deployment. The module Manage Configurations is important to help answer research question **RQ4** - *How can we evaluate the quality of implementation of the CMMI practices, ensuring that organisations fully attain their benefits and perform as expected?*, as the organisation can always compare its performance with different processes implementations and anticipate the impact of the changes introduced, even on the performance of processes other than the one being improved.

4.6 Repository Package

Looking at the Data Layer of EQualPI, in particular the Repository and its modules (see Figure 4.13), the Repository contains a **Data Dictionary** of the base and derived measures that are used to evaluate the quality of implementation of the practices. Organisation's performance data are stored in this database in the variables of the Data Dictionary and the way they relate with each other is given by the **Domain Model**. The Repository also includes the **Performance Indicators Models**, that are used to evaluate CMMI practices.

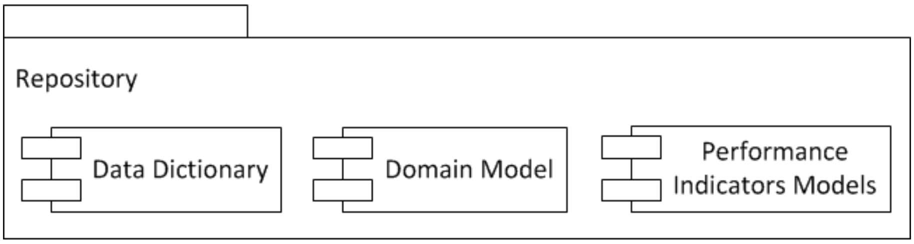


Figure 4.13: Contents of the Repository

The aforementioned modules are the ones to be used by organisations to implement and improve their practices when using the EQualPI Repository. The organisation's Data Dictionary has to define unambiguous variables in an understandable way, their measurement units, expected values and limits and the measurement protocol such that people collect them the same way. The Domain Model includes information needed to understand the relations between variables in the organisation context and contribute to the correct interpretation of the models and analysed data. Finally, the Performance Indicator Models are used for predicting and evaluating quality of the implemented practices. Such models can be built based on other organisations' data, but as the organisation begins collecting its own, its historic database is built. Therefore, the models shall be calibrated to use the organisation's own data.

4.6.1 Data Dictionary

To achieve better results organisations must have their data dictionaries defined and use them. The definition of the EQualPI's Data Dictionary was based not only on the effort estimation literature review (presented in 3.4 Effort Estimation) but also on the analysis of the TSP framework, as it includes several metrics that allow to plan, manage and control the entire software development process. The Data Dictionary is defined in a matrix where the rows are variables and the columns are elements used to define each variable, it can be found in (Lopes Margarido, 2012b).

When using TSP the software development teams collect data of several metrics, which are useful to fully characterise the software development process in terms of how the product was built, how effective it is and how efficient the development process was. Furthermore, by iteratively planning development through the PROBE method, using the necessary information to plan, and the TSP recommended values, contributes greatly to define more accurately the functionalities that the product will have, and all activities and required effort that are necessary to build, verify and validate the product.

In our research we determined the sources of TSP data by analysing them with the objective of defining the variables that are considered in the software development process. The Data Dictionary describes all variables recorded by TSP teams and supports the development of a TSP Database to store the data from any data source. Currently the SEI has a database of TSP workbooks provided by the practitioners. In January 2013, the database stored information of 257 workbooks. With the support of the Data Dictionary, the SEI extracted the data needed for this research from the workbooks database into a table, and validated them. The fact that the Data Dictionary is based on TSP metrics makes it useful to researchers that intend to analyse TSP data, because it fully covers the software development lifecycle, and useful to help TSP teams organise their data and know where to extract them from when they need to analyse such data. The use case of the TSP Database is presented in Figure 4.14.

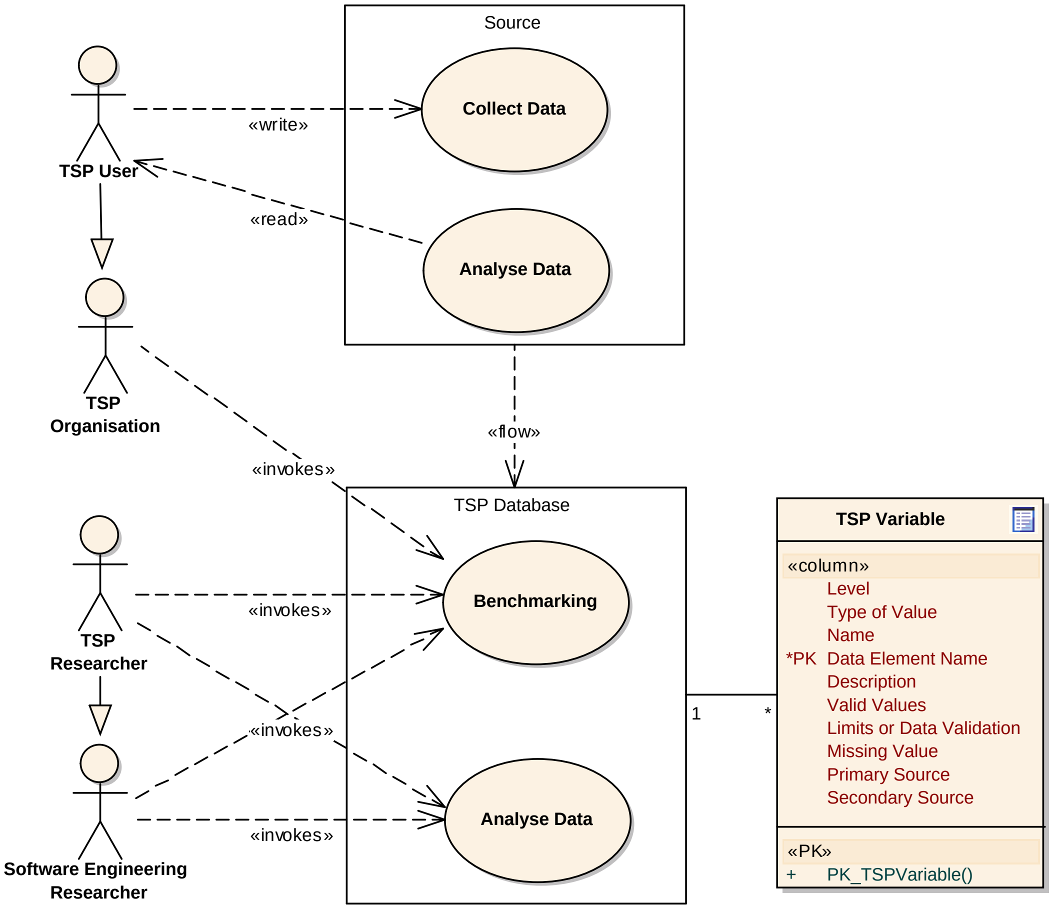


Figure 4.14: Use case of the TSP Database.

TSP data can be stored in the *TSP workbook* and *PSP workbook*, which are *Excel* files[1], or

[1] Obtained here: http://www.sei.cmu.edu/tsp/tools/tspi-form.cfm.

using the *Dashboard*[2]. Organisations can also have their own tools to store the TSP data, so for those organisations the Data Dictionary columns **Primary Source** and **Secondary Source** work only as a reference to remind them of which variables we are referring to. The data can also be stored in the following documents: *TSP Launch*, *Team Survey*, *TSP Postmortem*, *Quality Plan*, *Training Records* or in *Meeting 1: Management Presentation*, for example. The **TSP User** collects, compiles and verifies the integrity of the TSP/PSP data. The data are analysed during the project by the team, in the progress and post-mortem meetings, whereby each team member analyses their own data. The data are reused to plan subsequent project cycles and other projects.

The **TSP Database** includes data of projects from different organisations using TSP, who are the **Source** of the data. **TSP Organisations** can use the **TSP Database** data to compare their results and/or use it as historical data for estimation, in projects where they do not have their own. Researchers in general can use the data not only to conduct their research but also to compare their results.

4.6.2 Domain Model

We now describe the Data Dictionary, its variables and the relations between them, through the Domain Model, which we developed with the purpose of supporting the understanding of the Data Dictionary. At first, we analysed different schemas, such as the relational model for databases, and the star model for data warehouses, to define how the data of the variables of the Data Dictionary would be stored, but we concluded we could not impose a schema. Therefore, the Domain Model does not represent a relational model of a database, it was chosen as a representation useful to understand the variables at different analysis levels. Even though the Domain Model was designed with focus on extracting data from a TSP repository it can be adapted to get data from repositories of other software development processes. Together, the Data Dictionary and Domain Model, are necessary to collect the data and populate the Repository database with variables that will be used as Performance Indicators and in the Performance Indicators Models, to evaluate and anticipate the quality of implementation of the practices.

The Domain Model was defined at different levels of abstraction: classification, aggregation and generalisation (Neumayr et al., 2009). The first and highest abstraction level refers to the definition of the elements presented in the columns of the Data Dictionary, characterising each **Data Entry**, as defined in Figure 4.15. A Data Entry is a row of the Data Dictionary, that is a **Variable**, and has several properties that are represented in the columns: Type of Value, Level, Name, Data Element Name, Definition, Valid Values, Limits or Data Validation, Missing Value, Primary Source, Secondary Source, Role in Research, Type of Indicator, Process, Input or Output.

The Variable includes a **Name**, which is a logical name, a **Data Element Name** that is the unique name to designate that variable in the Data Dictionary, and a **Description** – altogether these columns are **Definition** elements. The **Constraint** elements that characterise the variable

[2] A tool that can be obtained here: http://www.processdash.com/.

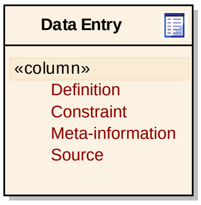


Figure 4.15: Data Entry elements.

are the **Valid Values**, **Limits or Data Validation** and **Missing Value**. The column **Level** is **Meta-information** used to characterise where the data are collected. Finally, the Data Entry includes information about the **Source** of the variable, which is where the data are normally stored. There are two columns for that purpose, the **Primary Source** that is the most common source for that variable, and the **Secondary Source** that is an alternative place to look for the data if they cannot be found in the primary source. We first defined the Data Dictionary based on projects and organisation's data that are collected during projects execution, then we complemented it with the data that are collected in TSP projects. Hence, the source of information is present in the Data Dictionary because we used TSP data to demonstrate the Framework and we had to know where organisations could get them.

Going down an abstraction level, entering in the details of Data Entry, we represent the structure of the Data Dictionary in Figure 4.16. Each **variable** will have a **time stamp**, a measurement **unit** and a measured **value**. The **valid values** of the variables are defined in the Data Dictionary. As they are necessary for the performance models, the variables are related to a process. The cases we have developed so far relate variables to the **estimation** or **development** process, of which the variables may be an **input** or **output**. Notice that the input of a process can be the output of another. As previously explained, the construct of a performance indicator, that is the **performance indicator model factors**, can be characterised by a set of **controllable** and **uncontrollable** factors. The variables play a **role in the model**, they can be **context** variables, hence, uncontrollable factors, and may also be **grouping** variables to allow aggregation. Context variables help to characterise the scenario in which the data collection is done. Another role that variables play is of **quality indicator**, consequently controllable factors. The quality indicators may be of **performance** (**efficacy** or **efficiency**) and of **compliance** with the process. This structure helps us answer to research question **RQ5** - *Is it possible to define metrics to evaluate the quality of implementation of CMMI practices focused on their effectiveness, efficiency and compliance?*

There are three dimensions to consider when using the Data Dictionary:

- **Time:** this dimension has different variables, depending on the level where it is being analysed.

- **Level:** this dimension considers where the data are collected/analysed.

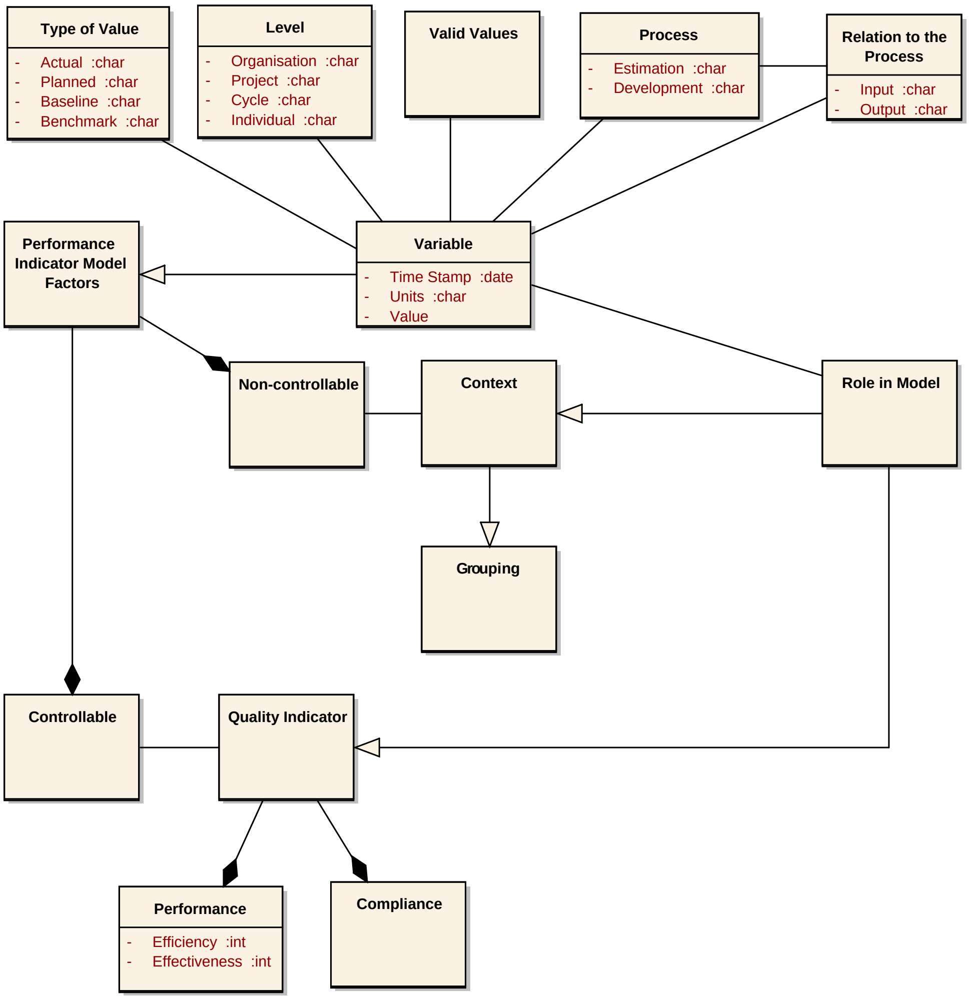


Figure 4.16: Structure of the Data Dictionary

- **Type of Value:** some variables have a suffix that indicates if the value is planned, baseline, actual or benchmark (_<Type of Value>). These suffixes are explained later in this section.

DataCollection_Period is a time dimension that exists at the **Level** *Organisation*, therefore it mainly characterises **Organisation_ID**. It may have more than one value and corresponds to the time interval when the projects/cycles, from which the data was gathered, occurred. This variable can have intervals of time when characterised by **CMMI_ML** and/or **TSP_Partner**, i.e. we can characterise periods of the organisation and expect that the results in projects are different if those periods have an influence on the results of the projects. A variation within the time period is also possible, in case the organisation improves the processes or lets them degrade. Even if the organisation is not rated with a CMMI level, nor is a TSP partner, the time interval is still relevant to characterise the usage of a technology, for example.

Start Date and End Date are related to several time variables such as the TSP launch (**TSP_Launch_Start**, **TSP_Launch_End**), the project or cycle itself, and in that case there are planned, baseline and actual dates: (**Start_Date_Planned**, **End_Date_Planned**, **Start_Date_Actual**, **End_Date_Actual**). **Baseline** also includes a start date and end date and refers to the period of time in which the baseline was applicable. This time interval is different from **Start_Date_Baseline** and **End_Date_Baseline**, which refer to the plan itself.

The Data Dictionary entries are analysed and aggregated at different levels. In Figure 4.17 we represent the metamodel of an organisation project, which is based on the SPEM2 (OMG, 2008) and the SPAGO4Q (Colombo et al., 2008) metamodel, and was adapted to better describe iterative projects of which spiral, prototyping, TSP or agile development models are examples.

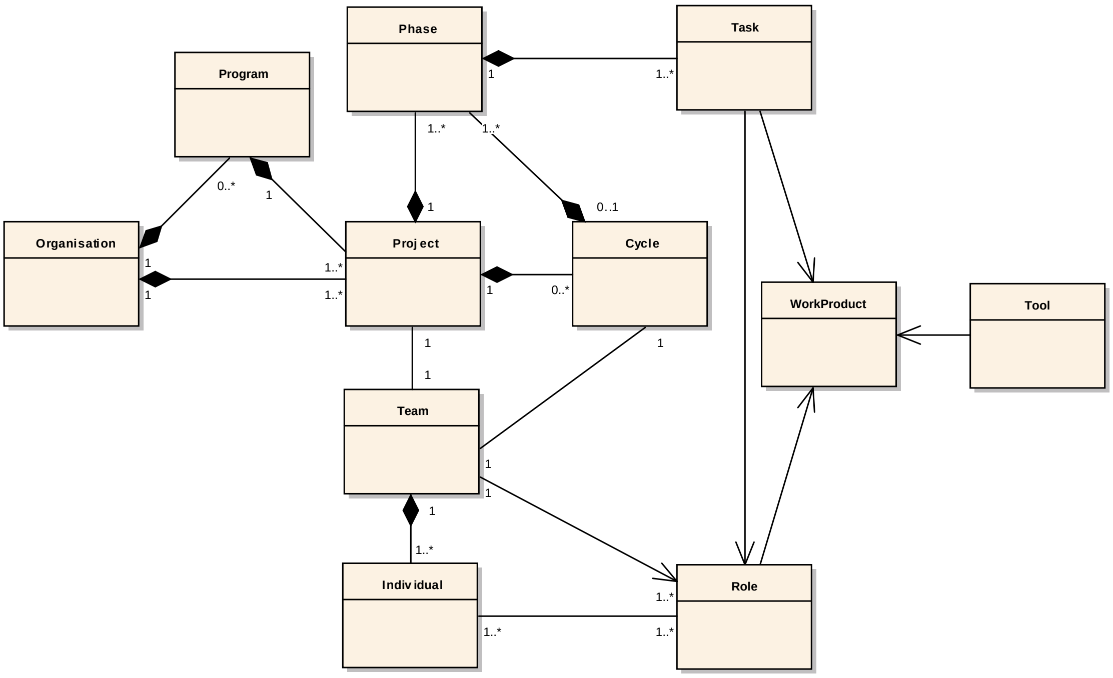


Figure 4.17: Iterative projects overview.

In a high level view, an **Organisation** has one or more **Projects**, whose development is comprised of one or more development phases. In certain cases more than one product can be developed through time in different projects, which are related to the same **Program**. In the case of applying TSP, for example, projects are done iteratively in cycles and each **Cycle** comprises one or more development phases. Each **Phase** includes a set of **Tasks** that have a **Role** associated, performed by one or more team members. Those tasks are necessary to develop the **Work Product** and **Tools** may be necessary to do the task. If the project was developed using Scrum (Schwaber and Sutherland, 2016) the cycle would be a *sprint* and the tasks to execute in a given cycle would be *items* in the *sprint backlog*.

Regarding the analysis of TSP metrics the data can be analysed at different aggregation levels (column Level of the TSP Data Dictionary): *Organisation*, *Project*, *Cycle*, *Team* or *Individual*. The individual data are aggregated in team data, team data are aggregated at the project level and the aggregation of projects' data helps to characterise the organisation. On the other hand, the team data can be used to characterise a project cycle. These are the levels where data can be collected and analysed. When using the Data Dictionary it is possible to filter the variables per Level and only analyse the ones that are at each one of the described levels.

The diagram in Figure 4.18 represents the variables that are exclusive of the organisation level. The Organisation has a *primary_key*, which is **Organisation_ID**. The **DataCollection_Period** represents the interval of time in which the organisation collected the data, however there may be more than one interval of time if there are different organisation characteristics for that period of time, namely regarding a CMM or CMMI maturity level and/or a period of time in which the organisation becomes **TSP_Partner** (yes or no). The **Business Goal**s may also vary through time or just remain unchanged over the time interval during which the data were collected.

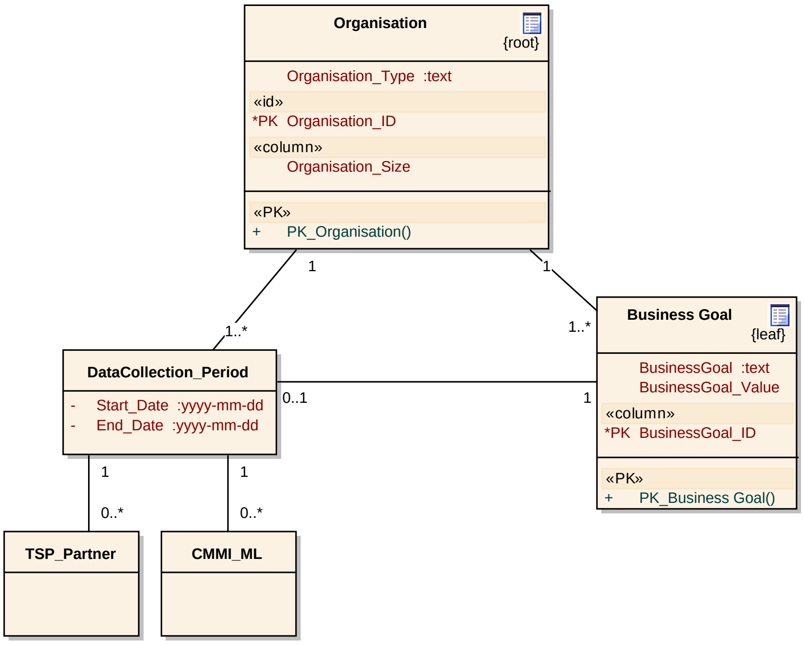


Figure 4.18: Organisation elements.

The variables in Figure 4.19 are related to the Project; however, they can also can be defined at Cycle level. A **Project** is developed for a **Client** or several clients and it includes a set of **Stakeholder Goals** that can be of the *Client*, different *organisation departments* or *upper managers*, and the *Team*. Any goal is characterised by a description **StakeholderGoal** and can have a target value **StakeholderGoal_Value** that the team tries to achieve. The **StakeholderGoal_Status** gives information as to whether the team met the goal or not and if the goal had a target value, the status includes the achieved value. **Project Goals** include the goals that were already specific to the project but also the goals that the team accepted along with additional ones it proposed, and to which it is committed; they can include the goals of the remainder stakeholders but can vary in coverage or target value.

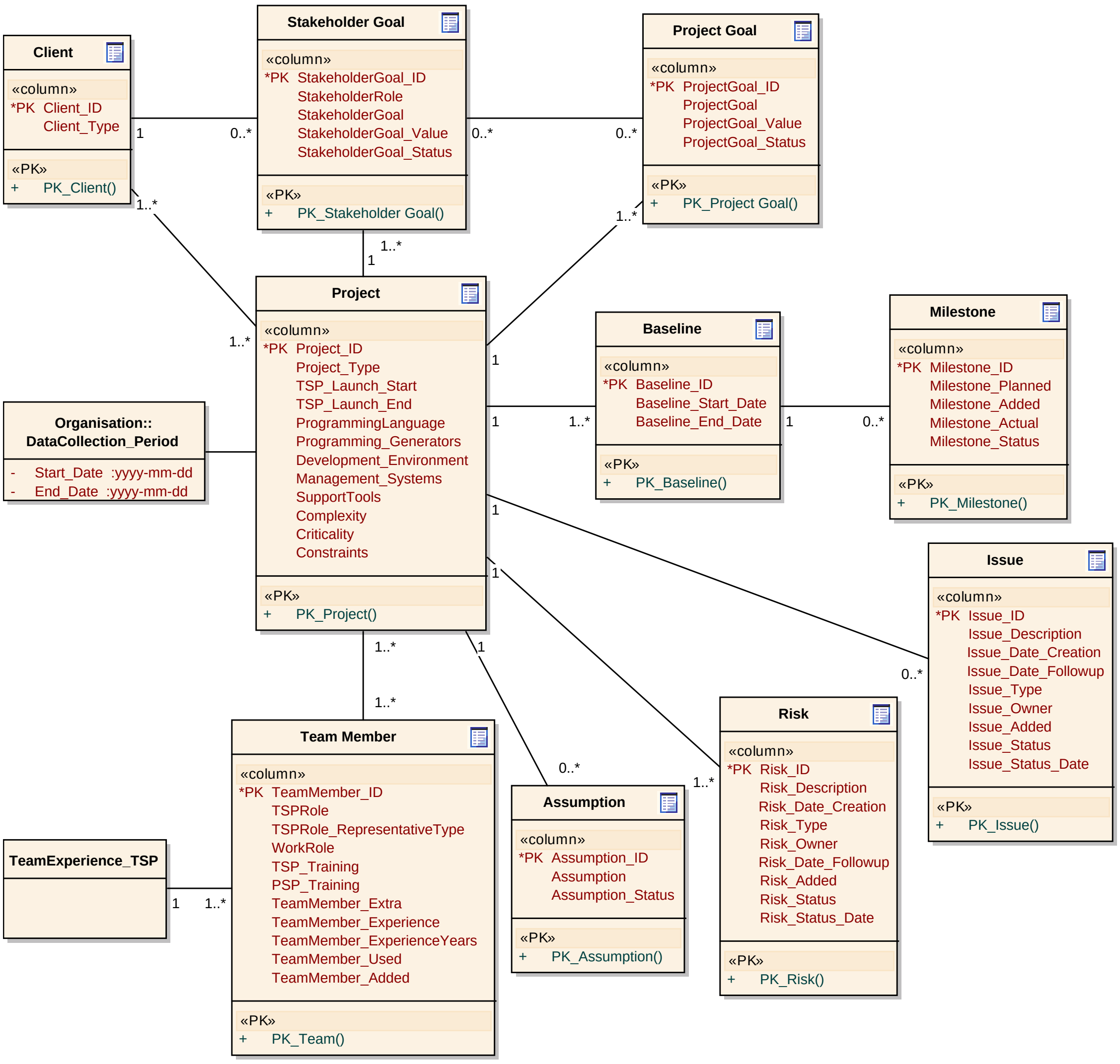


Figure 4.19: Project elements.

All projects have a set of attributes that characterise them. For example, in the particular case of the **ProgrammingLanguage**, it has meaning when related to the **DataCollection_Period**; the programming language can already be set in the market and be well documented and discussed by the community, or it can be in its early stage, where information is scarce and limitations are unknown.

A **Project** has a team assigned who executes it. In the particular case of TSP projects there are roles specific of TSP, **TSPRole**, that are assumed by team members, and have Primary and Secondary representatives, **TSPRole_RepresentativeType**. More generically, each member has a **WorkRole**, which determines what the team member does in the project (e.g. *Developer*, *Tester*, *Project Manager*). Not all TSP team members have TSP and PSP training and that is also identified. The difference between a **TeamMember_Extra** and a **TeamMember_Added** is that the extra team member only participates in the project in "at peak of work" circumstances and its participation is short in time. Added team members become part of the team either because their need was not foreseen at the beginning or a change in scope demands a bigger or more specialised team.

Often, **Assumptions** made by the team regarding the project are documented at the beginning of the project, **Risks** are identified and managed throughout the project and **Issues** are also recorded. They are all updated and managed over the project duration. Project plans have **Baselines** and may have **Milestones**, although in TSP many teams do not identify milestones. In the level *cycle* we identified several variables that are often compared with the baseline.

As mentioned before, in some software development lifecycles projects are developed in cycles that we represent in Figure 4.20. In case of TSP, each cycle begins with a launch meeting, where it is planned by the team and a post-mortem meeting, during which the cycle is analysed by the team and processes may be updated if necessary. The plan of a cycle is organised in weeks, where tasks are executed by team members in order to produce work products, here designated as **Program Elements**. Those program elements can be *Requirements*, *Detailed Design*, *Test Cases*, etc. or the *Code* itself; in this last case they are part of a **Component** which can be part of a **Module**. In the same way **Tasks** are assigned to **Team Members**, so are **Program Elements**. The development of a program element includes different **Phases** and is only complete when particular phases finish successfully. The **Phase** can refer to an introduction of defects phase, for instance *Coding*, *Detailed Design*, or a defects removal phase, such as *Inspection* or *Unit Testing*.

Certain variables in the Data Dictionary include **_Planned**, **_Baseline**, **_Benchmark** or **_Actual** extensions in their name. The "baseline", as previously mentioned, refers to the variable value in the current baseline. The "planned" is the value that is planned for the current cycle. The "benchmark" value is the value considered when planning the project to support planning and estimating, where the value can come from industry benchmarks, TSP Quality Guidelines Humphrey (2006), TSP recommended values for pilot teams, organisation historical data, expert judgement or changes to any of these benchmarks. The "actual" value is obtained when executing the process and should contribute to the growth of the historical dataset.

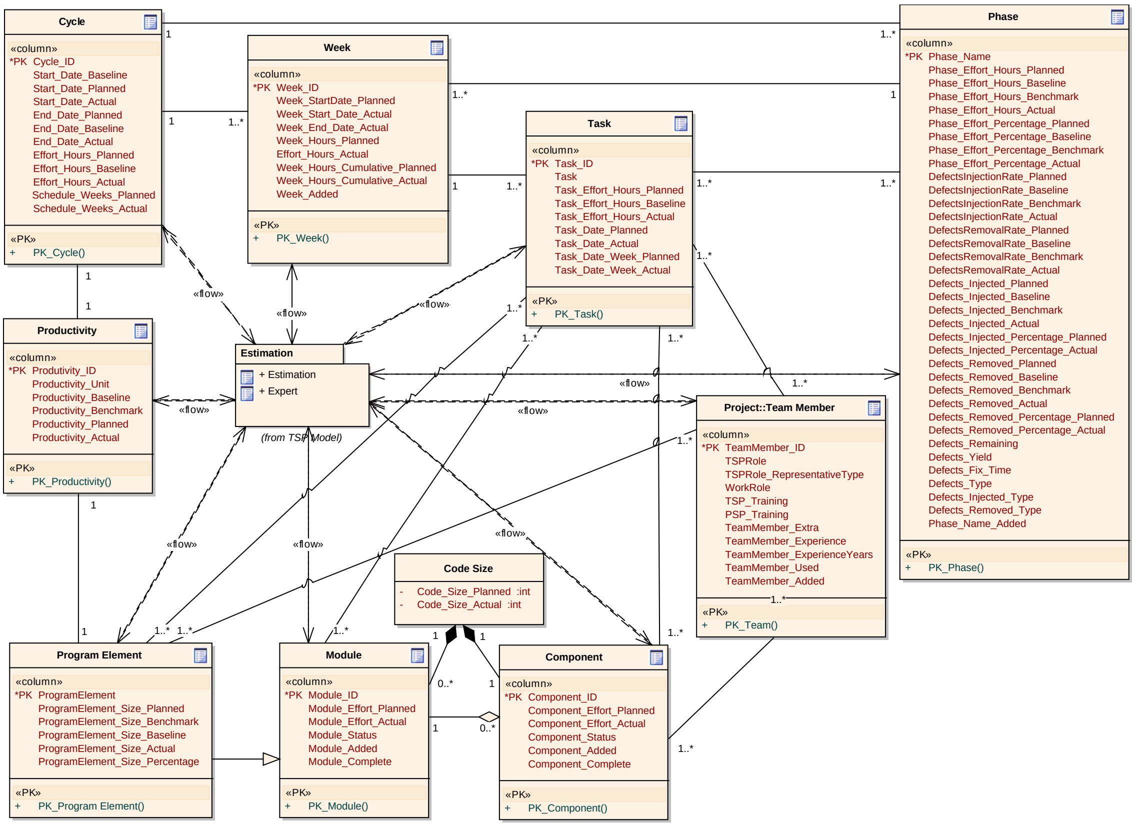


Figure 4.20: Cycle elements.

4.6.3 Performance Indicator Models: Effort Estimation Evaluation

The model of effort estimation uses the Data Dictionary variables. In this section we focus on the fields **Type of Variable**, **Process** (*Effort Estimation Process* or *Development Process*) and **Input** or **Output**. The following paragraphs describe how the Estimation and Development processes are related when considering the dependent variable (Y) Effort Estimation Accuracy.

The diagram in Figure 4.21 represents the estimation Process Variables. At the launch of the cycle all the components of the cycle are planned and the estimates of several variables are produced. The *Planned Values*, which are outcomes of the **Estimation Process**, feed the **Development Process**, which is also affected by external elements of *Context* and *Client Information*, for example. One of the outputs of the Development process is the *Actual Values*, as a consequence of how the process was executed, which can be used for appraisal and to feed the organisation database of *Historical Data*. Therefore, there is a bidirectional information flow between the Estimation process and all the components of the software Development cycle, because the estimates produced affect all planned values in development, and the actual values at the end of the development cycle, are considered for the estimation of the next cycle.

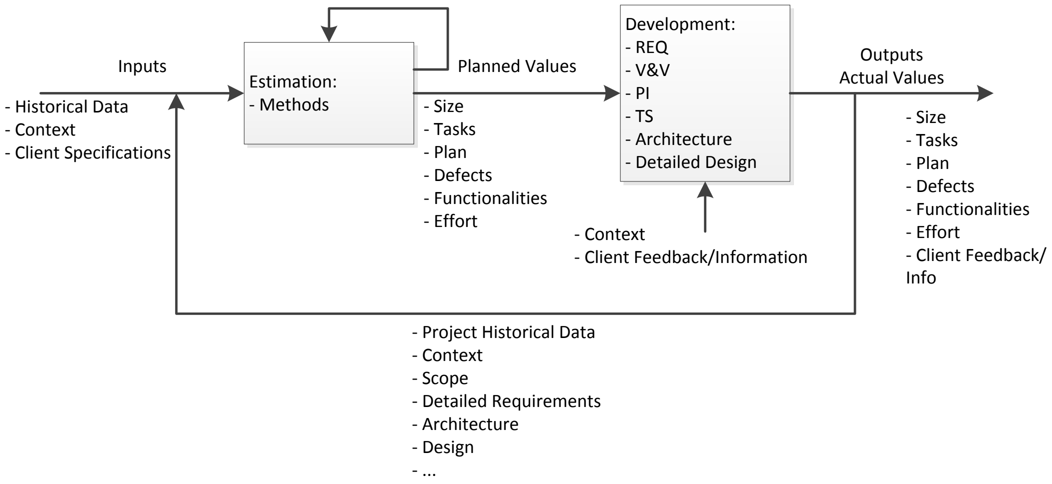


Figure 4.21: Estimation and Development processes feedback loop.

Legend: REQ - requirements, V&V - verification and validation, PI - product integration, TS - technical solution.

First Instantiation

When the effort estimation process is first instantiated there are a set of controllable and uncontrollable factors that need to be considered when analysing the performance of the process, and are included in the Data Dictionary. The controllable factors (X_c) are the variables on which we can act on, namely of process definition and execution:

- Choice of what is being estimated and considered to estimate effort.
- Choice of estimation methods to use.

Uncontrollable factors (X_{nc}) are variables that we cannot (yet) control, related with the project context and execution, complexity and environment:

- Context;
- Scope;
- What will happen during the development cycle.

To evaluate the quality of implementation of PP SP1.4 these two parcels, controllable and uncontrollable factors, need to be considered. If we determine the percentage of the effort estimation accuracy that each of these parcels represent we will know what the percentage is that we can control while estimating. To determine the effort estimation accuracy at a given point in time $t = i$ we can consider the deviation as being a partial variation (equation 4.5).

$$PartialEffortEstimationAccuracy: \quad EEA_t = \frac{Actual_t - Planned_t}{Planned_t} \tag{4.5}$$

The value of EEA_t is computed considering only the effort estimated for the tasks that were planned and executed between estimation moments, within a specific time period. The value can also be determined cumulatively across time periods, for example, from the beginning to the end of the project, as a global effort estimation deviation (see equation 4.6). The planned values are the ones from the first plan or proposal.

$$GloablEffortEstimationAccuracy: \quad EEA_{global} = \frac{Actual - Planned}{Planned} \tag{4.6}$$

The estimation process is instantiated throughout the project, as the uncertainty about it diminishes over time, the new knowledge must be considered in the project plan. It would be a mistake not to consider the new information to better define what needs to be done, how it will be done, the sequence of tasks and their duration, at the begining of a new development cycle. Ignoring such information to re-plan would be like ignoring information to build what the client truly expects and the understanding obtained from the technical knowledge that is gradually added. In development models that are not iterative, such as waterfall, the replanning can be done once requirements are approved with the client, and ideally, after validating a prototype. If the requirements change, the scope should be updated accordingly and, consequently, the plan should also be updated.

N^{th} Instantiation

The outputs and actual values of the Development Process at the moment of a n^{th} instantiation of the Estimation Process become a valuable input for it and include what part of the plan was already done. In other words, it is necessary to consider the actual values up to the current execution phase of the project. Tasks that were finished earlier than expected leave slots of time that can be used to execute other tasks, people that become available earlier can start following tasks or help other team members that are late to finish theirs.

Every output of the development process that adds detail to what is to be done and how it needs to be done (such as detailed requirements, architecture, design, scope), change requests to what has been done already and requirements that are added, changed or eliminated by the client – all imply re-planning. Another aspect is the number of defects that are actually being detected, which can trigger changes in the verification strategy, for example. By the analysis of complexity and execution of the task itself, it can be concluded whether it is necessary to add or remove effort in order to finish the task, or even add people or extra tasks, such as training.

Part of the uncontrollable factors of the first instantiation of the estimation process is considered in the n^{th} instantiation becoming controllable. The uncontrollable parcel only becomes more controllable if the project team can find and have the power to make them controllable (CMU/SEI, 2009, webinar video), otherwise, the team can just act on the controllable factors and consider the information that contribute to diminish uncertainty when re-planning. Then, there are as many re-planning cycles as there are development cycles. The objective of this approach is to benefit from the reduction of uncertainty to better and more accurately plan the next steps of the project. If re-planning is considered to set a new baseline, then the total effort estimation should consider the accuracy of each baseline, a variation that is a sum of the parcels deviation (see equation 4.7), and compare it with the global deviation previously defined.

$$TotalEffortEstimationAccuracy: \quad EEA_{total} = \sum_{t=1}^{k} EEA_t = \sum_{t=1}^{k} \frac{Actual_t - Planned_t}{Planned_t} \tag{4.7}$$

The procedure we followed to define performance indicators models is based on the GQiM[3]. The questions we want to answer when evaluating a practice are:

- What is the purpose of the practice?
- What do we want to achieve with it?
- How do we know we achieved what we wanted?

This was the principle we followed to define which performance indicator we would use to evaluate the Effort Estimation process. The answer to the questions is we want to *build a project plan with effort estimates that are close to the effort we actually need to execute the project*. If we were analysing the process Requirements Specification, our intention with the process would be to *define requirements that translate what the customer intends to have*.

Then the question to ask is: What indicator can be used to evaluate whether that goal is achieved? In our case, was the *effort estimation accuracy*, in order to know how close to the actual effort was to the estimated one. In the case of requirements would be *how the feature was close to the requirement*, which can be evaluated by checking *if the corresponding acceptance test passed*. So we would get an *acceptance percentage* traced with requirements to evaluate the *requirements degree of fulfilment* of the customer needs.

Lastly, it is necessary to ask: What variables can be analysed to build a model that helps to understand the deviation of the process actual from the expected behaviour? In the case of effort estimation, what *process steps* and *influencing factors* can we measure and analyse to know if they are part of our model or not.

We build the EEA model in section 5.3 Evaluation of the Estimation Process, which helps to better understand the procedure to define a Performance Indicator Model and answer research

[3] See section 2.3 Process Performance Measurement and Improvement.

question **RQ4** - *How can we evaluate the quality of implementation of the CMMI practices, ensuring that organisations fully attain their benefits and perform as expected?*

So far, we presented the EQualPI framework's overview, architecture and detailed several of its modules, as can be seen in Figure 4.22. The Framework is based on the researchers experience, discussions with other researchers from around the world, including experts from the SEI, literature reviews of prior work, and qualitative and quantitative research. The problems and recommendations gathered, to help organisations implement CMMI, resulted from a literature review, further analysis of data of a survey conducted by the SEI involving organisations that were appraised at HML, and three case studies we conducted on organisations rated at CMMI level 5. The recommendations of process improvements were based on the researchers' experience in the software engineering industry, experiments conducted with students and adoption of the process improvement we designed, as well as validation by an organisation. The Process Indicator model provided with EQualPI was generated through the analysis of SEI TSP data.

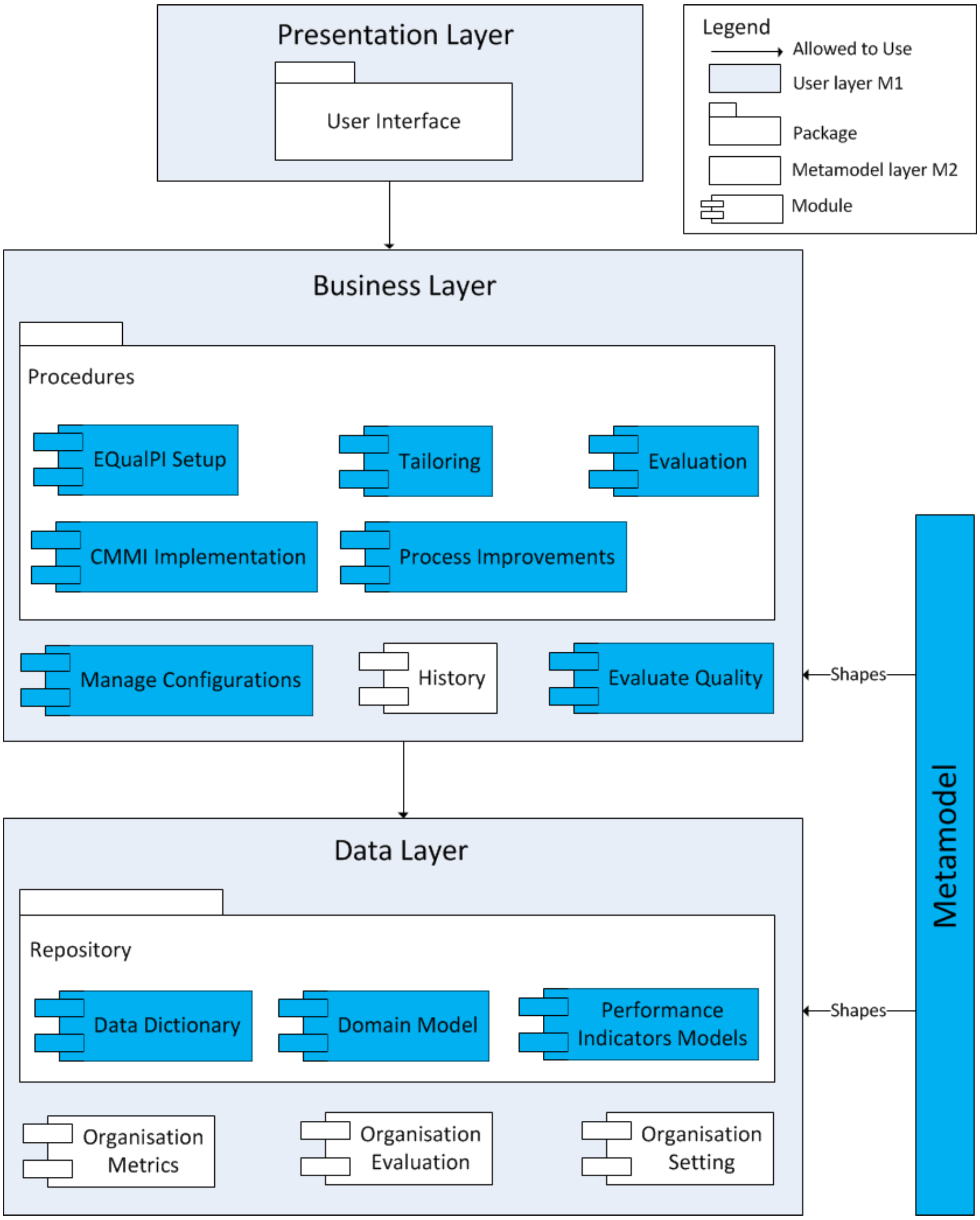


Figure 4.22: Modules of EQualPI that we defined signalled in blue/shadowed, including the metamodel.

Chapter 5

EQualPI Validation

In this chapter we present the validation of EQualPI and part of its modules. We defined and validated the Framework based on the literature reviews, using metamodelling and data modelling, conducting case studies, conducting data analysis on surveys and organisations' projects, performing quasi-experiments and building regression models using TSP data. In all our statistical analysis we considered a confidence interval of 95%, α or significance level of 0.05.

In section 5.1 CMMI HML Implementation, we validate EQualPI's CMMI Implementation conducting three case studies in organisations appraised at CMMI level 5 that helped complement the list of problems and recommendations provided in EQualPI. We also did further analysis of the data gathered on the SEI surveys in organisations aiming to achieve high maturity to complement, validate and add variables that influence, contribute to, or are used by, organisations that achieved the CMMI high maturity goal.

In section 5.2 Requirements Process Improvement, we validate part of the steps defined in EQualPI to conduct process improvements by doing the requirements review process improvement introducing a defects classification specific of requirements. The defects classification was adopted by a CMMI level 5 organisation.

In section 5.3 Evaluation of the Estimation Process, we validate the EQualPI's Effort Estimation Evaluation model by analysing TSP variables data to select Performance Indicators of the estimation process and build a model of Effort Estimation Accuracy (as defined in equation 4.6). To extract the TSP data from the database, we and the SEI, used the EQualPI Data Dictionary, that indicated which variables were needed, their definition and where they could be found if not in the database. Additionally, the Domain Model was also useful to understand better the relations between variables and define how to aggregate them to different granularity levels (tasks, components and projects).

5.1 CMMI HML Implementation

This section refers to the validation of the EQualPI's procedure CMMI Implementation, presented in 4.4.2 CMMI Implementation: Problems and Recommendations. The Procedure and the theory of EQualPI are based in our experience, the literature reviews we did in several areas, including the one presented 3.1.3 Problems in Process Improvements, Metrics Programs and CMMI, and while conducting a new case study in organisations appraised at HML, completing a total of three. Therefore the definition and validation of the CMMI Implementation procedure, adjustment and enrichment of EQualPI and the particular case of this procedure was an iterative process, contributing to building and evolving the Framework in time.

Our focus on HML was motivated by the challenges faced when implementing PPM and PPB in organisations that are moving from lower maturity to higher maturity. We wanted to identify factors that are relevant to achieve HML, which lead us to further analyse the data of two surveys that the SEI conducted in organisations aiming to achieve HML, as we presented in 3.2 Survey on MA Performance in HML Organisations, and improve the list of problems and recommendations (checklist in 4.1). We analysed and statistically tested the relations between several of the survey variables and the achievement of the HML goal, and build valuable PPMs and PPBs, respectively.

Along my professional career, I had been using CMMI and was part of an appraisal team to achieve CMMI level 5. I felt the difficulties of implementing the model and found several gaps, which made the CMMI journey also a discovery. Similarly, in our research we found some of the problems we were expecting in the literature review (see 3.1.3 Problems in Process Improvements, Metrics Programs and CMMI) but also additional ones. We mapped them with recommendations to help organisations overcome or avoid those difficulties. Furthermore, our personal experience showed us differences in the implementations and results achieved by different organisations, motivating us to further research the problems and challenges that the organisations face when implementing CMMI, in particular ML 4 and 5, and help them improve their results. In fact, another key problem is that measuring organisations performance is outside SCAMPI scope; the assessment verifies if the techniques applied allow achieving CMMI goals (Masters et al., 2007). There are only two PAs where performance improvements are explicitly analysed: CAR, where the effect of implemented actions on process performance should be evaluated; and OPM, where the selection and deployment of incremental and innovative improvements should be analysed (CMU/SEI, 2008). Given the SEI concerns about high maturity and actions taken to ensure organisations understood high maturity, along with the SCAMPI limitations presented in section 3.1.4 SCAMPI Limitations, we conducted three case studies in multinational organisations that develop software (CI, CII and CIII) assessed at CMMI for Development ML 5, staged representation. Case studies in CI and CIII were conducted immediately after the SCAMPI A appraisal. We expected to find difficulties and even some unstable practices and performance models, in those two particular cases.

The results of these case studies, the literature review that we did to find the problems in CMMI, metrics programs and process improvements, and the analysis of the data of the SEI surveys, taken in conjunction, allowed us to build and validate the CMMI implementation checklist (see 4.1). We mainly focused on MA and HML, but also analysed other CMMI PAs. The research questions we intended to answer, which we considered when designing the case studies and analysing all data, were the following:

- What was the strategy to evolve to the new ML?
- What difficulties and problems occurred in the implementation of the new practices?
- What is the process definition?
- How was the process defined?
- How are people using the process?
- How are people collecting, analysing and interpreting process data?
- What is the impact of the new process on people's work?

When considering HML organisations, we expect they show they understood CMMI and we can see that they did the following:

- Implemented the necessary processes to achieve the CMMI goals and present evidence of using those practices, additionally they get the expected performance results.
- Fulfil CMMI practices having no gaps (according with the SCAMPI rules).
- Being High Maturity organisations they have PPM and PPB in place that represent their processes and projects and use them in projects and process improvements.
- Conduct innovation projects based on their performance and ensure that the implemented improvements do not cause the degradation or erosion of other processes that are presenting good quality results and show balance of cost, efficiency and effectiveness along with compliance.
- When implementing the practices people doing the work are involved, reflecting organisation culture, having processes that are adequate to the organisation's activity and are meaningful to the people using them.

5.1.1 Further analysis of the HML Survey Data

To further enrich EQualPI's CMMI Implementation module and the list of problems and recommendations that is part of the Procedures package, we further analysed the data of the 2009 survey, (presented in 3.2 Survey on MA Performance in HML Organisations) supporting the problems we have identified and the recommendations to overcome them. These are part of the EQualPI's Procedures we presented in 4.4.2 CMMI Implementation: Problems and Recommendations. We analysed the graphics of relations between some of the practices that were questioned in the survey and achieving HML (or not) by the respondent's organisation and complemented our analysis with statistical tests to verify if the differences found were significant.

The survey question to ask organisations if they gave monetary incentives or not in recognition of the effort of people involved in MA initiatives included a "check all that apply" question in the

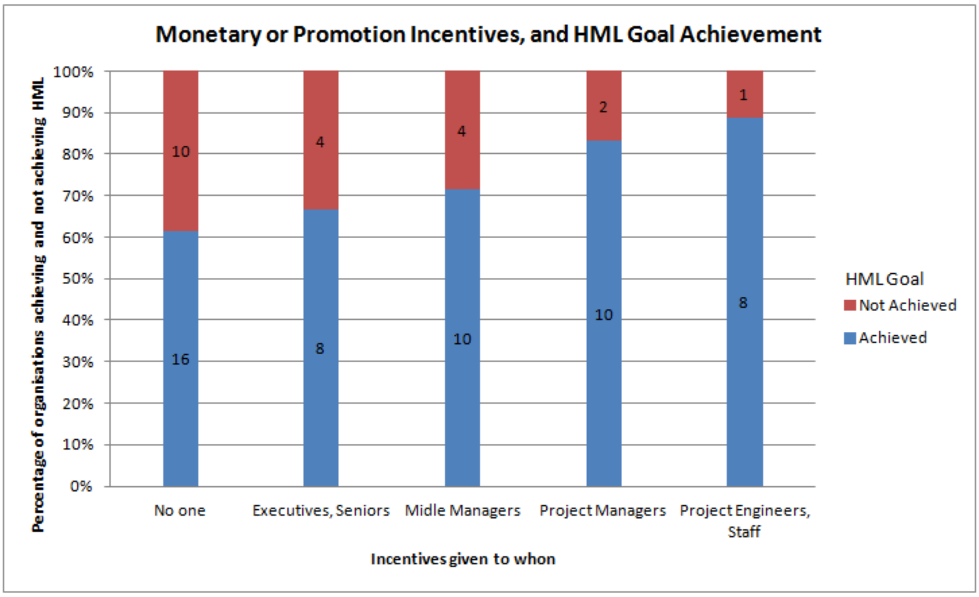


Figure 5.1: Relationship between giving incentives to people who use and improve MA, and the achievement of the HML goal (Lopes Margarido et al., 2013).

case of giving monetary incentives. We represented the information in a graph that is presented in Figure 5.1 to analyse if the organisations that achieved HML gave monetary incentives and, when given, to whom those incentives were given. The first column indicates that a larger percentage of organisations that achieved HML did not give promotions or monetary incentives, which could tell us that such incentives do not influence the achievement of HML. Nonetheless, looking at the data of the organisations that give those incentives we can see that people at lower levels in the organisation structure, who are working closer to the projects (Project Engineers, Technical Staff and Project Managers), are more likely to receive incentives when working in High Maturity organisations.

The OPP practices are essential to build the information needed to be able to understand organisations' processes current behaviour and predict how they will behave in the future. That is why PPM and PPB are so relevant to achieve high maturity. The survey included a question of how well the managers who use PPM and PPB understand their results, as misinterpretation of process data may lead to wrong conclusions and inadequate decisions being taken. Analysing the relation of this understanding with the achievement of CMMI HML, looking at the graph in Figure 5.2, we notice that in HML organisations managers tend to understand the results well and extremely well as opposed to the managers of organisations that did not achieve high maturity, who instead tend to have a moderate or worse understanding.

The organisations were asked about how well the creators of PPM and PPB understood the intent of CMMI with PPM and PPB. We expect that the results on the scales used in understanding CMMI definition of PPM and PPB, and when to use them, are higher in HML organisations. Figure 5.3 shows that a larger percentage of organisations which achieved HML understood either

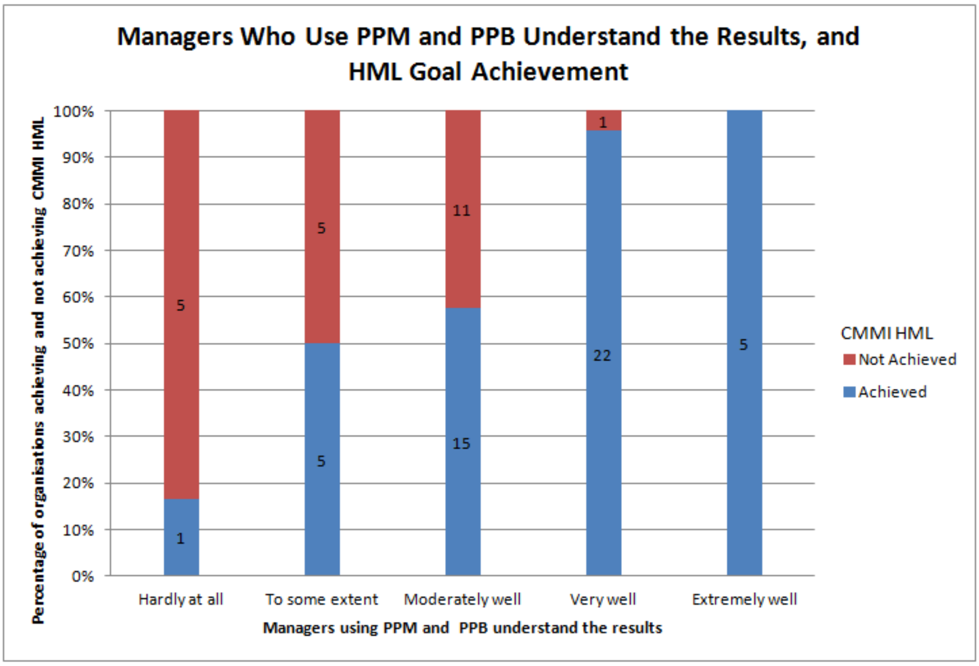


Figure 5.2: Relationship between managers who use PPM and PPB understanding the obtained results, and the achievement of the HML goal (Lopes Margarido et al., 2013).

very well, or extremely well, the definition and usefulness of PPM and PPB. This is an indicator of the importance of having relevant training for people creating and giving support to their usage, because it is as important to understand the definitions as it is to understand when they must be used, and ultimately how to use them.

As we observed the importance in achieving HML of having managers who understand the results of PPM and PPB and also of creators who understand the CMMI PPM and PPB definition and their usefulness, we drew the graph in Figure 5.4 to see if we could detect a relation between these two variables. What we see is that managers understand well, or extremely well, the results they are analysing when the creators of PPM and PPB also understand their definition better and also understand when they must be used.

With the previous result in mind, and given that one of the survey questions is about "qualified, well-prepared" availability to work on process performance modelling, we also created a graph of how well managers understand PPM and PPB and the availability of the experts to work on PPM. This is presented in Figure 5.5. We see that managers tend to understand the results well, or extremely well, when experts are available almost always.

In TR2008, Goldenson et al. showed that the relationship between the overall value of PPM and checking data quality and integrity was strong. We analysed the relation of the various methods to the achievement of the HML goal (Figure 5.6) and some of the graphs indicate that integrity data checks also seem to be related to the achievement of HML. When looking for cases where more than 75% of organisations which had followed the practice achieved HML, while less than 30% still achieved HML without having followed the practice, we see that "distinguish missing

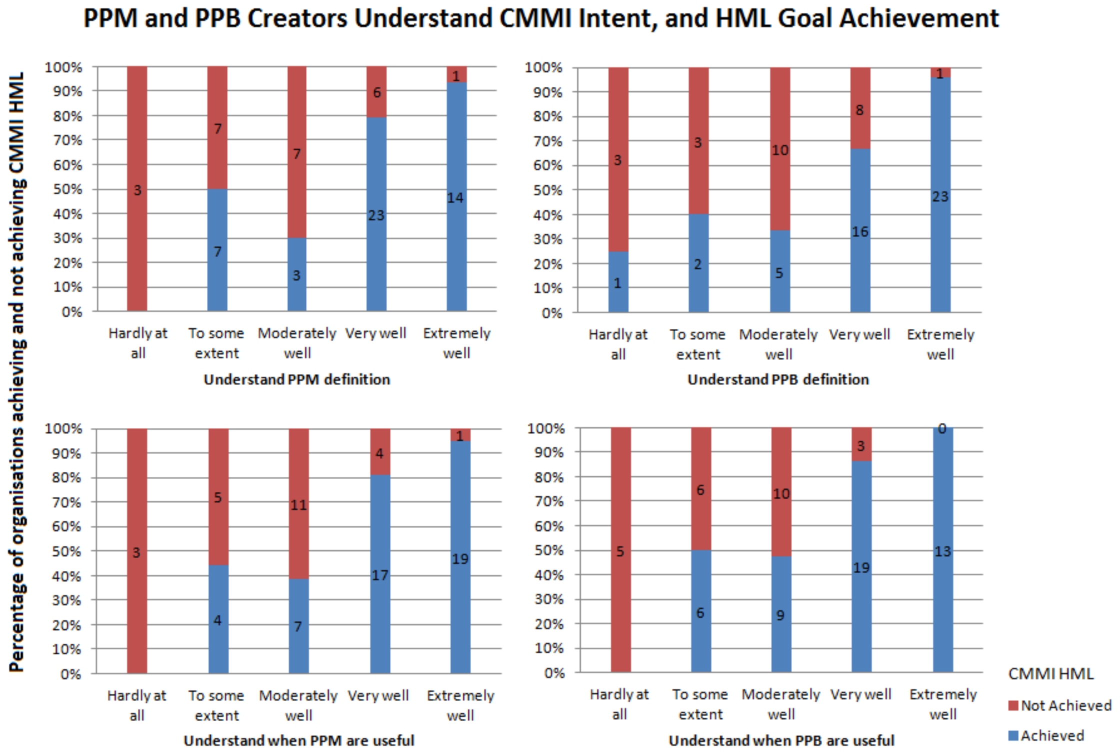


Figure 5.3: Relationship between understanding the CMMI intent with PPM and PPB by their creators, and the achievement of the HML goal (Lopes Margarido et al., 2013).

values from zeros" and "check unusual patterns" follow these criteria. Looking at the remaining data integrity practices to identify the cases where approximately 75% of the organisations which had executed the practice achieved HML while less than 50% still achieved HML but did not use the practice, we find the practices "check out of range and illegal values", "use data precision and accuracy" and "estimate measurement error" to be of relevance.

For statistical confirmation of what we observed when analysing the graphs, we used the statistics described in Table 5.1, allowing us to understand the relationship between achieving HML (V1) and doing a given practice. We tested the dependency of several variables and compared the groups of organisations that achieved HML with those which did not. In Table 5.2 we report the tests results of dependent variables (p-value < 0.05 in the Chi-square test) with different central tendency between groups (p-value < 0.05 in the Mann-Whitney test, used to test the medians of ordinal variables). The Levene test is used to ensure that the subjects in the sample come from the same population (p-value >= 0.05).

The variance of the groups is generally similar; as expected, the medians are different, showing that the level of understanding of CMMI intent with PPM and PPB and when they are useful, availability of experts to work in PPB, and executing the mentioned data checks and integrity practices, influence the dependent variable; and there is in fact an association between variables. We were unsurprised to have confirmed the relation between V1 and how well managers under-

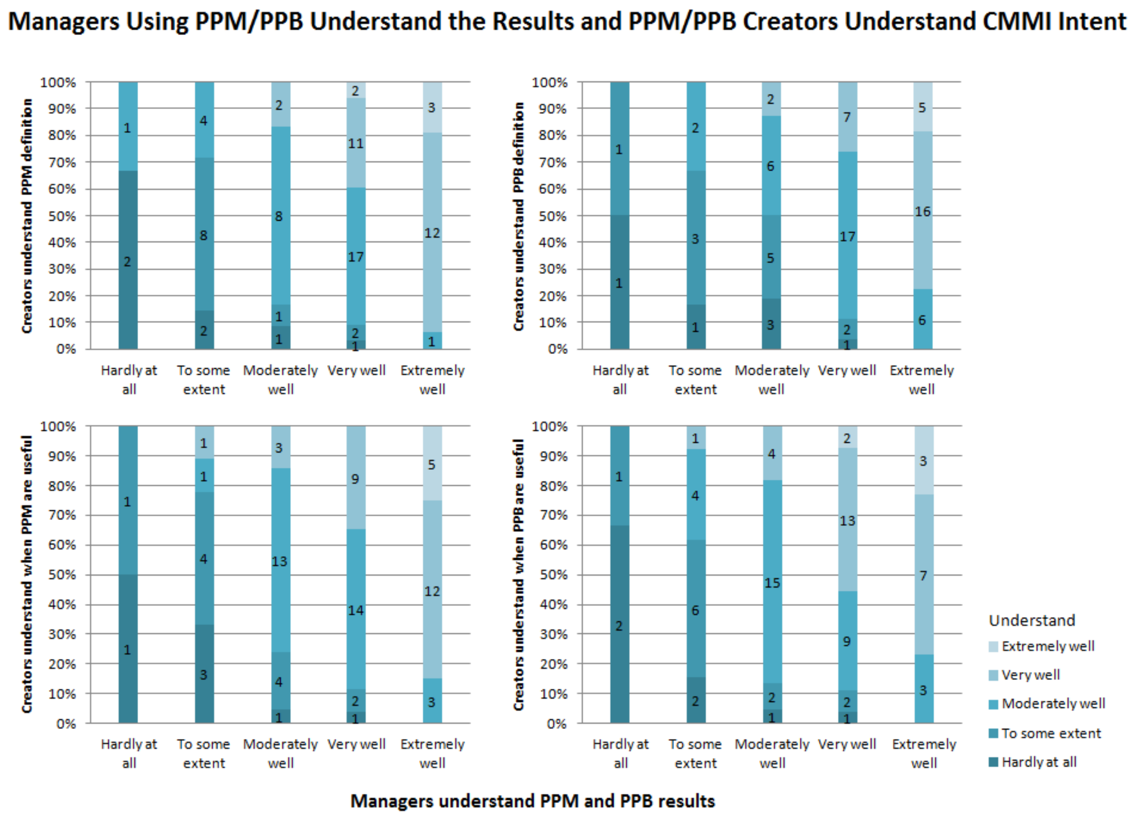


Figure 5.4: Relationship between PPM and PPB creators understanding the CMMI intent and managers who use them understanding their results (Lopes Margarido et al., 2013).

stand PPM and PPB results. This also indicates they are making informed decisions to achieve the intended results, and benefiting from using PPM and PPB. Nonetheless, the quality of the information depends on the quality of PPM and PPB, so consequently the result obtained may also be affected by that quality. PPM and PPB must be useful and based on reliable data. We also verified that there is a relation between the understanding that the creators of PPM have of the CMMI (V4 to V7) and achieving HML.

In both surveys the relationship between understanding results of PPM and PPB, by Managers who use them (V2), and the achievement of HML was quite strong. We observed that there is a relation between the prevalence of managers who understand the results better (V2) and creators who better understand the CMMI intent (V4 to V7). This result may be an indicator that the PPM and PPB on those cases are built considering CMMI requirements and those tend to be understood well. Another relation that we analysed and confirmed within the statistical tests was that managers tend to understand PPM results better when experts are available to work more often (V8). Such result was to be expected as, in my experience, building and understanding PPM and PPB when implementing CMMI was a challenge due to the complexity of concepts when moving from LML to HML.

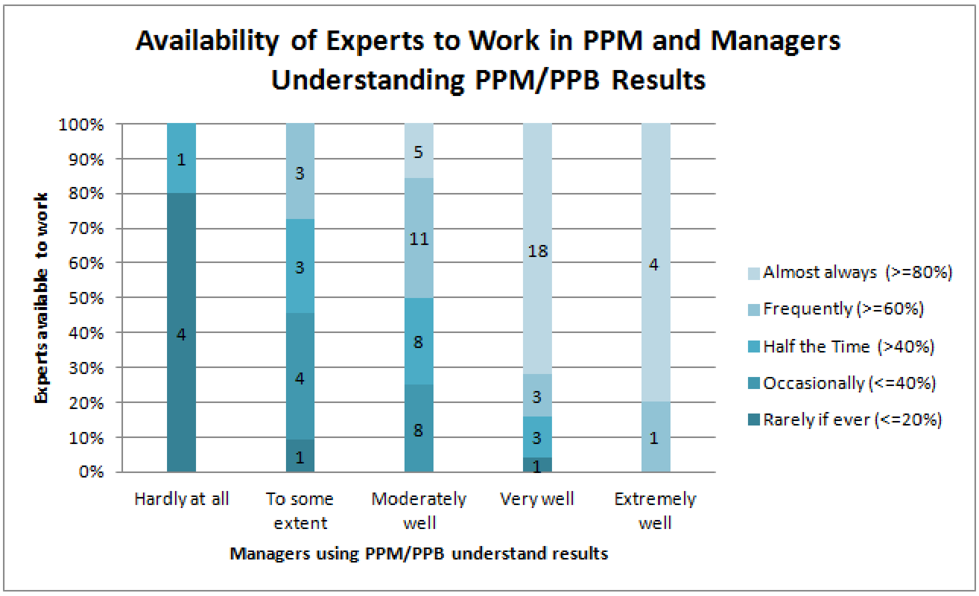


Figure 5.5: Relationship between the availability of experts to work in PPM and managers who use them understanding their results (Lopes Margarido et al., 2013).

Table 5.1: Statistics and hypotheses that were tested.

Test	Description	Hypotheses
Levene	**Purpose:** Verify conditions to use the Mann-Whitney U test. The samples must have the same variance. If p-value < 0.05 we can reject the null hypothesis: the variance of the two samples are different. If p-value >= 0.05 keep the null hypothesis: the variances of the two samples are the same.	H_0 (Null Hypothesis) – the two samples (Achieved and Not Achieved) have the same variance H_1 (Alternative Hypothesis) – the two samples have different variance
Mann-Whitney	**Purpose:** Verify if the two samples have similar median. If p-value < 0.05 we can reject the null hypothesis: the median of the two groups is different. If p-value >= 0.05 there is no difference between the groups.	H_0 – the two samples have similar median H_1 – median of Achieved > median of Not Achieved
Chi-Square	**Purpose:** Verify dependency between two variables. If p-value < 0.05 we can reject the null hypothesis: the variables are dependent. If p-value >= 0.05 the variables are independent.	H_0 – the variables are independent H_1 – the variables are dependent

The statistical tests support that, of the data integrity checks that the survey included, distinguishing missing data from zeros (V9), checking data precision and accuracy (V10) and estimating measurement error (V11) are related to achieving HML. Please note that V11 could only be considered by increasing the confidence interval from 95 to 99%, and hence changing the p-value to consider that the groups have the same variance (verified using the Levene test) to be >= 0.01.

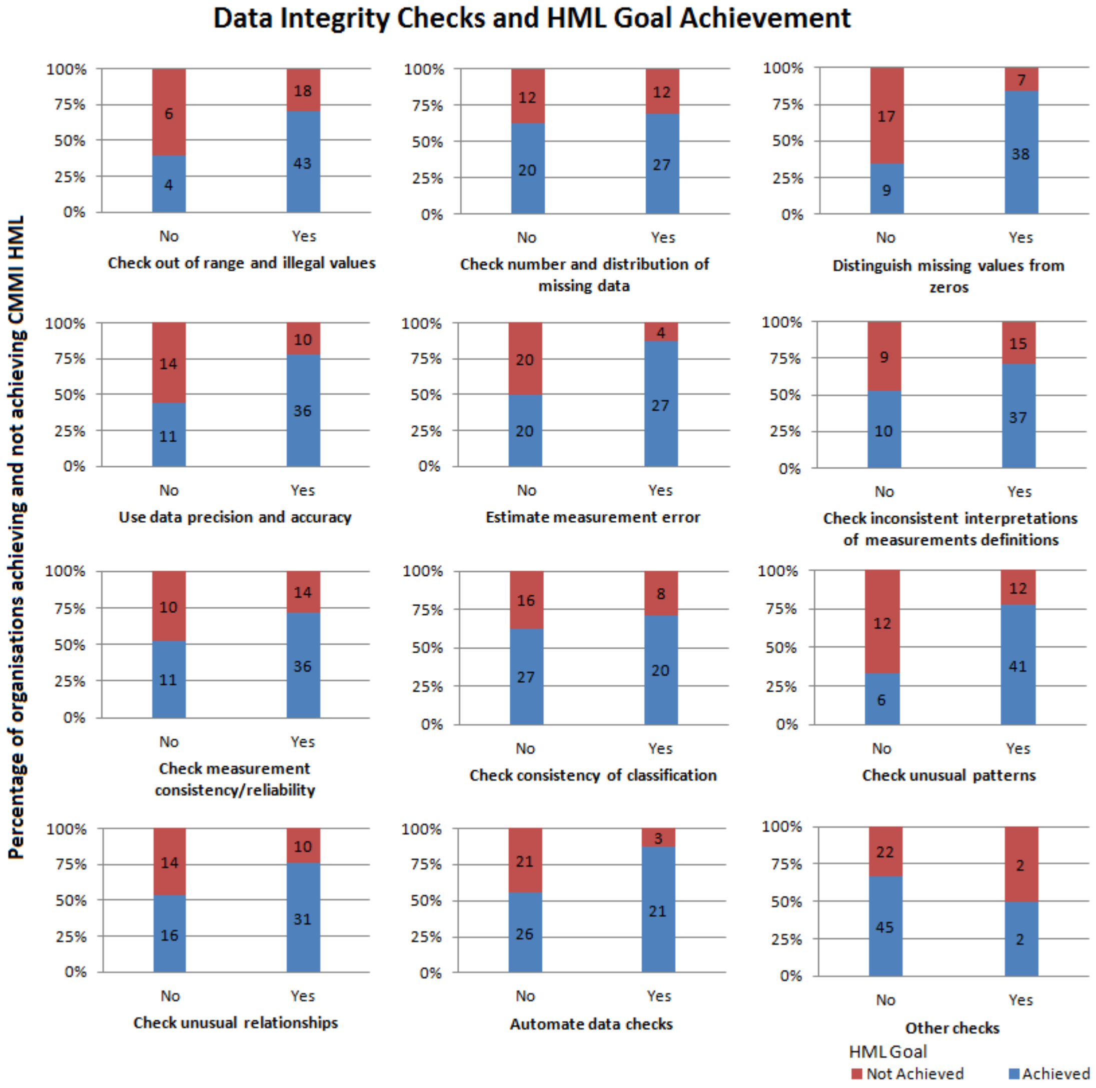


Figure 5.6: Relationships between performing data integrity checks and the achievement of the CMMI HML goal (Lopes Margarido et al., 2013).

Following the analysis and tests we performed we found that the following variables are related to the achievement of the desired HML:

- How well managers understand PPM and PPB;
- How well PPM and PPB creators understand the CMMI intent;
- Distinguish missing data from zeros;
- Check data precision and accuracy;
- Estimate measurement error.

Table 5.2: HML 2009 survey – further data analysis (Lopes Margarido et al., 2013).

Variables	**Levene**	**Mann-Whitney U**	**Chi-Square**

V1: CMMI HML goal Achievement
V2: How well managers understand PPM and PPB results
	F = 0.0399
	p-value=0.8423
		W = 200.5
		p-value = 1.44e-05
			X-square = 20.647
			p-value = 0.000372

V1
V4: PPM and PPB creators understanding of the definition of PPM given by CMMI
	F = 1.4462
	p-value = 0.2333
		W = 875
		p-value = 7.75e-05
			X-square = 20.537
			p-value = 0.000391

V1
V5: PPM and PPB creators understand the definition of PPB given by CMMI
	F = 0.0484
	p-value = 0.8264
		W = 902.5
		p-value = 1.71e-05
			X-square = 19.644
			p-value = 0.000587

V1
V6: PPM and PPB creators understand when PPM are useful
	F = 0.0082
	p-value = 0.9281
		W = 920
		p-value = 7.74e-06
			X-Square = 23.235
			p-value = 0.000114

V1
V7: PPM and PPB creators understand when PPB are useful
	F = 0.1445
	p-value = 0.7051
		W = 931.5
		p-value = 4.20e-06
			X-square = 24.846
			p-value = 5.40e-05

V2
V4,5,6,7: PPM and PPB creators understand CMMI intent
	F = 3.1665
	p-value = 0.07963
		W = 576.5
		p-value = 0.02413
			X-Square = 9.8091
			p-value = 0.020261

V2
V8: Availability of experts to work in PPM
	F = 0.0095
	p-value = 0.9227
		W = 163
		p-value = 1.05e-05
			X- Square = 23.211
			p-value = 0.000115

V1
V9: Distinguishing missing data from zeros
	F = 0.8992
	p-value = 0.3463
		W = 855.5
		p-value = 2.25e-05
			X-square = 16.126
			p-value = 5.93e-05

V1
V10: Checking data precision and accuracy
	F = 2.5641
	p-value = 0.1139
		W = 761
		p-value = 0.00389
			X-square = 7.0344
			p-value = 0.007996

V1
V11: Estimating measurement error (CI 99%, p-value = 0.01)
	F = 4.9559
	p-value = 0.02927
		W = 794
		p-value = 0.001166
			X-square = 9.1471
			p-value = 0.002491

Note: results of the tests done with the groups of organisations which achieved, or did not achieve, HML and which were shown to have the same variance through the Levene test.

These relations reinforce the importance of doing proper data integrity checks in order to have meaningful and reliable PPM and PPB supporting high maturity PAs. Furthermore, to ensure that managers understand PPM and PPB results correctly, and consequently take appropriate actions, PPM and PPB creators must understand their CMMI meaning and usefulness, and experts must be available. We complement the works of Goldenson et al. (2008) and McCurley and Goldenson (2010) with our findings, considering all of them recommendations that should be considered to build valuable models and achieve HML (see Table 5.3, ours are signalled in bold). Our results were integrated in the EQualPI's list of problems and recommendations to implement CMMI, identified with SDA (Table 4.1).

Table 5.3: MA recommendations for HML (Goldenson et al., 2008; McCurley and Goldenson, 2010; Lopes Margarido et al., 2013).

Purpose	Recommendation
Building Valuable Models	Put emphasis on "healthy ingredients"
	With purposes consistent with "healthy ingredients"
	Diversity of models to predict product quality
	Diversity of models to predict process performance
	Use statistical methods: regression analysis for prediction, analysis of variance, SPC control charts, designs of experiments
	Data quality and integrity checks
	Use simulation or optimisation methods: Monte Carlo simulation, discrete event simulation, Markov or Petri-Net models, probabilistic models, neural networks, optimisation
Factors related with HML CMMI goal achievement	Have good documentation relative to process performance and quality measurement results
	Use simulation/optimisation techniques
	Have diverse methods of simulation/optimisation
	Have models with emphasis on "healthy ingredients"
	Have models for purposes consistent with "healthy ingredients"
	Substantially use statistical techniques
	Regularly use PPM in status and milestones reviews
	Managers must understand well PPM and PPB, which is related with the following three
	PPM and PPB creators must understand the PPM and PPB definition given by CMMI
	PPM and PPB creators must understand when PPM and PPB are useful
	Have experts available to work in PPM
	Distinguish missing data from zeros
	Check data precision and accuracy
	Estimate measurement error

With this research, we were able to identify additional factors that are useful to implement CMMI, but are also relevant evidence to look for, when appraising an organisations. Therefore, we contribute to answer research questions **RQ3** - *What additional recommendations can we provide to organisations to help them avoid problems when implementing CMMI?* and **RQ4** - *How can we evaluate the quality of implementation of the CMMI practices, ensuring that organisations fully attain their benefits and perform as expected?*

5.1.2 Case Studies

Building and validating EQualPI's procedure to implement CMMI was an iterative process beginning with the literature review of existing problems in CMMI and conducting case studies over the years, to list and consolidate some of those problems and recommendations and adding new ones. We focused on how organisations prepare for SCAMPI when implementing CMMI to achieve a maturity level and wanted to find strategies that could be used to avoid the problems we were finding, and of those methods, determine which ones were actually used by the analysed organisations during the implementation of CMMI. In this section we signal the problems found in each organisation with CI, CII, a Business Unit of the CII Group (CIIG), or CIII, respectively and the ones found in the analysis of the survey data with SDA. The problems (P) and recommendations (R) are numbered as they were when presented in section 4.4.2 CMMI Implementation: Problems and Recommendations.

In CI we interviewed the CMMI programme sponsor, posing direct questions and an open-end question which the interviewee would answer by narrating the story of the programme. We had a similar interview with the programme responsible. In each interview we identified our next interviewees, projects, tools and further documentation to analyse. We analysed the company Quality Management System (QMS) and PPM and PPB in use, Information Systems (project management, data collection, data analysis and product management tools) and SCAMPI A repository. We also interviewed practices and tools implementers, and project teams, including the appraised ones (whose documentation we analysed). Analysing CI data we found that the main problem stemmed from rapidly evolving to ML5 without giving enough time to have stable tools, processes, PPB and people behaviour.

CII is a business unit, located in several countries, that is part of CIIG, a CMMI level 5 organisation. In CII we interviewed the person responsible for the CMMI programme, beginning with direct questions and finishing with descriptive questions regarding the story of the programme. Afterwards we analysed CIIG QMS, PPB and PPM, CII procedures, and Information Systems (project management and data analysis tools). Finally, we had a meeting with the CMMI programme responsible and discussed our results and conclusions. In CII we found several problems related to metrics; most limitations stemmed from the fact that size was not being measured, and time spent on tasks stopped being accurately collected. Many of the identified problems were originated by a resistance to change and difficulty in presenting, to CIIG, metrics adequate for CII.

In CIII we interviewed a consultant involved in the appraisal of the organisation, i.e. a person who performed an actual observation on the case. The main difficulty faced by CIII was to move to statistical thinking.

When we prepared the interviews questions we had a set of categories where the answers would fit, based on the problems we knew from our experience, and the ones found in the literature:

- Entry conditions;
- Process definition and implementation divided in CMMI process areas;
- Metrics definition;
- Processes and metrics usage;
- Tools implementation;

- Tools setup;
- Training;
- PPM and PPB usage;
- Data analysis.

In all case studies we had similar higher level categories of history of the programme, difficulties found, benefits and achievements. In the programme history we found how the different organisations overcame the problems they faced. We conducted the interviews individually, to make people comfortable to share reality, rather than what they could consider to be the expected answers. We also asked the respondents permission to record the interviews, indicating the answers would be kept confidential, so as we asked the prepared questions we could note down the facts we would like to further explore and ask additional questions about. We transcribed the interviews data to an excel file organised by the identified categories and additional ones we found in patterns of similar problems and solutions. The time to transcribe 20 minutes of an interview was around 1 hour. We also extracted relevant information from the documentation and information systems. This approach allowed us to group the answers and observations into the same categories. Within the difficulties faced, we identified different categories of problems and mapped the answers/documentation information that fitted the same categories, resulting in the categories and groups of problems we have in the CMMI Implementation checklist (Table 4.1). We cross-checked the sources of information and also further investigated the few contradictions we found, in order to understand how the processes were designed and were actually being used. The final validation of results with the programme responsible gave them a better view of organisation performance and how the team got to those results. This approach also allowed us to confirm our findings. The problems found and solutions applied in these organisations are indicated in the next paragraphs.

Entry Conditions

P1. Underestimate time to implement HML

We found this problem in CI and CIII. According to the programme sponsor and responsible, CI re-planned the CMMI implementation programme several times at first until finding the right tools to implement the additional practices. This also was due of needing time to implement and see changes take effect. The programme sponsor stated that the first plan was unrealistic, as they were finding on their own how to move from ML 3 to ML5 and only then was it realised they needed a framework to base their metrics on and treat data, and thus they needed a tool to allow them to collect the data of new metrics. That was when Six Sigma was recommended by the Lead Appraiser and introduced in the organisation, which allowed them to better understand HML demands. With the Six Sigma practices the base processes implemented thus far were kept but a full new layer was added on top, to collect additional metrics and build the performance models and process baselines. Along the project the programme manager also felt the difficulties of having few resources and the most valuable people being requested to work in critical projects, not dedicating the planned time to CMMI. Only once the upper management set that achieving CMMI level 5 as a priority, the project went back on track.

CIII also had an unrealistic plan, following the interview responses from the consultant. The implementation turned out to be more complex than anticipated and the programme took longer than planned.

R1. Plan time for all necessary process improvement activities

In CIII the implementation plan was long and all activities needed to be executed on the estimated time. At first, they considered that if an activity had overrun the schedule, time could be recovered by shortening others. In reality, this was abandoned once they realised that time could not be shortened in other tasks.

P2. Introduction of HML forgetting ML 2 and 3

This problem occurs often and CIII was no exception, as there was a focus on ML5 without having a mature level 3 implemented and institutionalised.

R2. Have mature and stable levels 2 and 3

CI analysed gaps to address problems in lower maturity levels[1] because they had already been established for a few of years but had not evolved in line with organisation growth (interview with the programme sponsor). Besides, those processes were affected by changes to implement HML and there should have been a new cycle for them to mature. In CIII the move from ML3 to ML5 was uninterrupted, so the base was not yet mature. Regardless, with the move to ML5, ML3 matured and did not erode in the meantime.

P3. Understand the statistical/quantitative nature of level 4

This problem was found in CI and CIII. A move to statistical thinking and quantitative management was the main challenge faced by CI, it finally occurred when starting using Six Sigma. Even so, the definition of processes and tools, to use and analyse the new data was still being stabilised by the time of SCAMPI A.

Changing mentality to HML was a significant shift for CIII, because preparing and using the quantitative component takes time to mature. This problem may have been one of the causes of P1.

R3. Involve statisticians with experience in software and CMMI

In the SDA we showed the relations between achieving CMMI HML goal and having PPM and PPB creators who understand both their definition given by CMMI and when they are useful. Additionally, the statistical tests we carried out also revealed the relation between how well managers understand PPM and PPB results and the availability of experts to work.

[1] New projects to implement the new PAs were also in progress and SCAMPI C resulted in a set of projects to address gaps in several PAs.

R4. Introduce Six Sigma

In CI, Six Sigma helped to gain insight of information needs to achieve quantitative goals, solve problems and design PPB and PPM.

R3 to *R5. Top down and bottom up goals review*

Were also followed by CI and CIII, as they were part of the CMMI implementation process.

Process Definition and Implementation

P4. Processes copied from the CMMI model

We found a few cases of process in the CIIG QMS that were the same as described in CMMI, including specific practices used as steps but which did not reflect the organisation's culture. That may have been one of the reasons for the problem *P5. Multicultural environment.*

R6. Processes reflect organisation's culture and *R7. Involve experts and users of the processes*

Both these recommendations were followed by CI and CIII first by understanding the existent process, identifying gaps and then involving internal experts and users in the definition of improvements and new processes.

P5. Multicultural environment

CIIG was a large multicultural corporation with BUs spread around the rkrld, including CII. People from different cultures have different ways of working, following orders and displaying discipline. Also they have different ways to deal with change. While in certain cultures orders are taken without question, in others people need to understand the benefits of working in a certain way, otherwise they will resist change.

R8. Promote processes sharing and lessons learnt amongst different BU teams

CII applied this recommendation in a Business Area with specific needs. More specifically, they analysed other business units metrics in order to adopt the ones that could be applicable to their projects lifecycle.

P6. Impose processes

CIIG acquired other companies and imposed its processes on them; consequently their good practices, certain metrics and good visibility of processes were often lost. The imposed process may have caused P5.

R9. Goals specific to different organisation levels, related to the organisation's business goals and *R10. Monitor at different report levels*

This approach helped CI maintain the visibility of processes and projects at different organisation levels. These recommendations were part of the CMMI implementation process in CIII.

P7. Dissemination problems

This problem happened in CI, where several team members of appraised projects indicated that some of the processes were introduced without prior training or without enough detail to execute the practices, one of these being data analysis and integrity checks. This problem contributed to their difficulty in using them and in some cases, for example reporting effort of certain activities and classifying defects, they used them incorrectly for some time. The manager of the processes support tools projects also recognised that there was a delay in making them available and giving information of how they worked. Even after providing general training, people would still have more detailed questions when using them. By the time the appraisal occurred people noticed communication improvements; new ways of disseminating the information were available and part of them included the common questions of usage. Nonetheless, the dissemination of information regarding processes and tools usage was not totally effective: some people still had difficulties applying the new practices and part of the information was still not available to all projects teams.

R11. Commitment from the entire organisation, *R12. Complete and adequate training* and *R13. Projects and people coaching*

These recommendations were used by both CI and CIII. Regarding the recommendation R12 we cannot be sure it was effective in any of them. Additionally, at least in one of the CI project's the coach corrected people's mistakes, rather than guiding them towards correct behaviour.

P8. Lack of institutionalisation

The problem occurred in both CI and CII. In CI not all project teams were applying the new practices. According to a team member, many of other projects were not yet using some of the practices. Members of other projects also indicated that some of the new practices were not applicable to their projects. This problem was also related to people's behaviour and resistance to change.

In CIIG not all projects and business units performed at the same maturity level, according to the programme responsible. The differences found between the documented process of CIIG and the practices used in CII also revealed this problem.

R14. Top management: set goals, plan, monitor and reward

The recommendation was used by CI and CIII, where the dissemination of processes was gradual, as they were ready to be deployed directly from pilot projects to the entire organisation. However, when organisations are large they should consider even more gradual dissemination, spreading practices in a small group of projects and gradually involving new ones, which can be done in order to also profit from team members mobility. In the SDA we showed that the incentives to people improving and working in MA were more frequent in organisations that achieved HML.

R15. Mature processes and metrics with practice

Was used by CIII, in the progressive processes implementation and by letting the practices mature.

Metrics Definition

P9. Meaningless metrics

This problem occurred in CI: we found a case of a PPB of effort per phase being misinterpreted due to lack of understanding of the context of one business area. Similarly there was a metric of effort of executing unit tests, when in some projects that effort only took few seconds and thus negligible.

P10. Metrics definition (collect and analyse data)

This problem occurred in CI and CII. In CI people still had difficulties in collecting data in certain contexts and in their interpretation. That was mainly the case of effort reports while testing and fixing defects, and classifying defects. When first starting to compare the project data with the baselines the responsible for the task found several difficulties. Even the way certain data points were to be eliminated as outliers, when for the team they were justified and real, was still a difficulty.

CIIG imposed KLOC as the applicable size metric, which CII did not consider adequate to its types of projects. Some of the other business units used their own size metrics, one of them was in fact more suited for CII.

R16. Measurement reflecting goals

The recommendation was used by both CI and CIII so there was a clear view of which metrics were used to monitor different levels of goals and what their definition was.

R17. Measurement and analysis protocols

Was also followed by CI but definitions needed to mature to ensure unambiguous collection and interpretation. In CIII it was necessary to define new metrics for ML5 so as to have the desired confidence, as the integrity of existent data from ML3 could not be assured. With time the definition of metrics was improved to tune the process models.

P11. Uncorrelated metrics

This occurred in CIII at start, which implied conducting new data collection cycles and new searches for correlations. This may have been one of the causes of P1.

R21. Gradual data collection

CIII finally realised that to get meaningful data and be able to draw sustained conclusions they needed to do trials and, once they had a good base, collect data progressively.

P12. Metrics categorisation

This problem was found in CI. The data for high maturity had been collected for a short period of time, so the baselines were not stable enough. It was not possible to distinguish between different categories of data (representing the various markets, team experience, team sizes and project sizes), therefore the data were compiled in PPB categorised by technology only.

P13. Baselines not applicable to all projects

This was a problem common to CI and CII. In CI PPBs were still unstable, or inadequate for all types of projects. According to one team member, and what was said by others regarding the same process, one of the performance models depended on the level of expertise of the executers, and that variable was not considered. Moreover, time to collect data was insufficient to gather information with different contexts and verify if:

- New metrics were needed;

- There were differences in performance and in which contexts;
- In certain circumstances the procedure to collect the data should be different.

CIIG had centralised PPBs not applicable to all business units' realities and projects. For example, the development lifecycle phases had different durations depending on the business. Another problem was that productivity was measured considering a business day as unit of time, but some locations had different business day durations in number of hours.

The recommendations that could help avoid P12 and P13 were not followed by any of the organisations of the case studies.

Metrics Usage

P14. Abusive elimination of outliers

In CI we found one situation when it was not perceived that outliers of time to fix a defect that took longer than usual, in the examples 1 and 3 days, occurred at least once in some development projects. There was also a case of rejecting a code review results and asking the team to repeat the process, when the team had actually better performance in code reviews because team members were more thorough implementing and debugging before submitting code to review.

R25. Recognise special causes of variation and *R26. Quarantine outliers that are not understood*

In the SDA the tests we performed showed a statistically significant relationship between achieving HML and checking data precision and accuracy.

P15. Data not being collected in all projects

This problem occurred in CI and CII. In CI tools were not yet prepared to collect data in certain projects (maintenance, with several phases or outsourced), because their data structure was different from the standard projects (development). The only data collected represented 25% of the organisation's projects. Besides, measurements specific to maintenance and outsource projects were not defined.

As CII was not using KLOC to measure size they were not using many of CIIG's derived measures that were based on it.

P16. Effort estimates

While in CIIG effort estimation was based on their historical data of effort and size, CII estimates were based on expert judgement without using any tools or models.

R29. Use related historical data

Was followed by CI when first estimating time spent in similar tasks of comparable projects, for example.

R30. Build historical database by planning iteratively

CI followed the recommendation in TSP pilot projects.

P.17 People behaviour

This problem happened in CI and CII. In CI changing mentality was a challenge; some people did not see value in new practices or stated that they were not applicable to their projects. It is difficult to convince people to report effort accurately if they normally do not report effort as they are finishing tasks because they consider that it causes them to lose focus, and so they leave the reporting until later. Some of the practices still involved manual loading of files, which discouraged people.

CII workers stopped reporting effort accurately, and only reported contractual hours of work.

R31. Never use personal data to evaluate people

CI did not use personal data for evaluation purposes. Even following R31, CI had difficulties to convince people to accurately report effort – that is why we suspect that showing the benefits of training may not have been totally effective.

R32. Data quality and integrity checks

The results we obtained in the SDA showed that organisations achieving the HML goal:

- Distinguish missing data from zeros;
- Check data precision and accuracy;
- Estimate measurement error.

These practices allow to detect when variables and their measurement error are not presenting the expected behaviour.

Tools Setup

P18. Tools setup and requirements stability

In CI the existing Information System evolved to support the new practices, but people were still detecting problems and requesting improvements, in tools and processes, where this resulted from them using the tools in practice and in different projects contexts.

R33. Improve tools with usage

In both CI and CIII the tools initially used were more rudimentary. As processes, metrics and performance models and baselines were defined, more complex tools were adopted or implemented.

P19. Overhead

This problem happened in CI. Tools were not completely integrated and people were inexperienced in the new practices. To report effort spent on tasks people manually filled a form per task and the data collection of the new metrics was only partially automated. This problem was also due to the insufficient Tools Setup time, to understand all needs with the usage of the tools.

R36. After PPM and PPB stabilisation only collect necessary data

The experience in CIII was that initially it was necessary to collect data of all variables they felt could be important to create models and establish baselines. In time the non-used metrics were abandoned, leaving only the truly necessary ones.

R37. Use automated imperceptible data collection systems

The recommendation was followed by both CI and CIII. However, it is always difficulty to totally eliminate human intervention to report effort, especially when people have other tasks than just developing code, for example.

R18. Appropriate metrics and *R17. Measurement and analysis protocols*

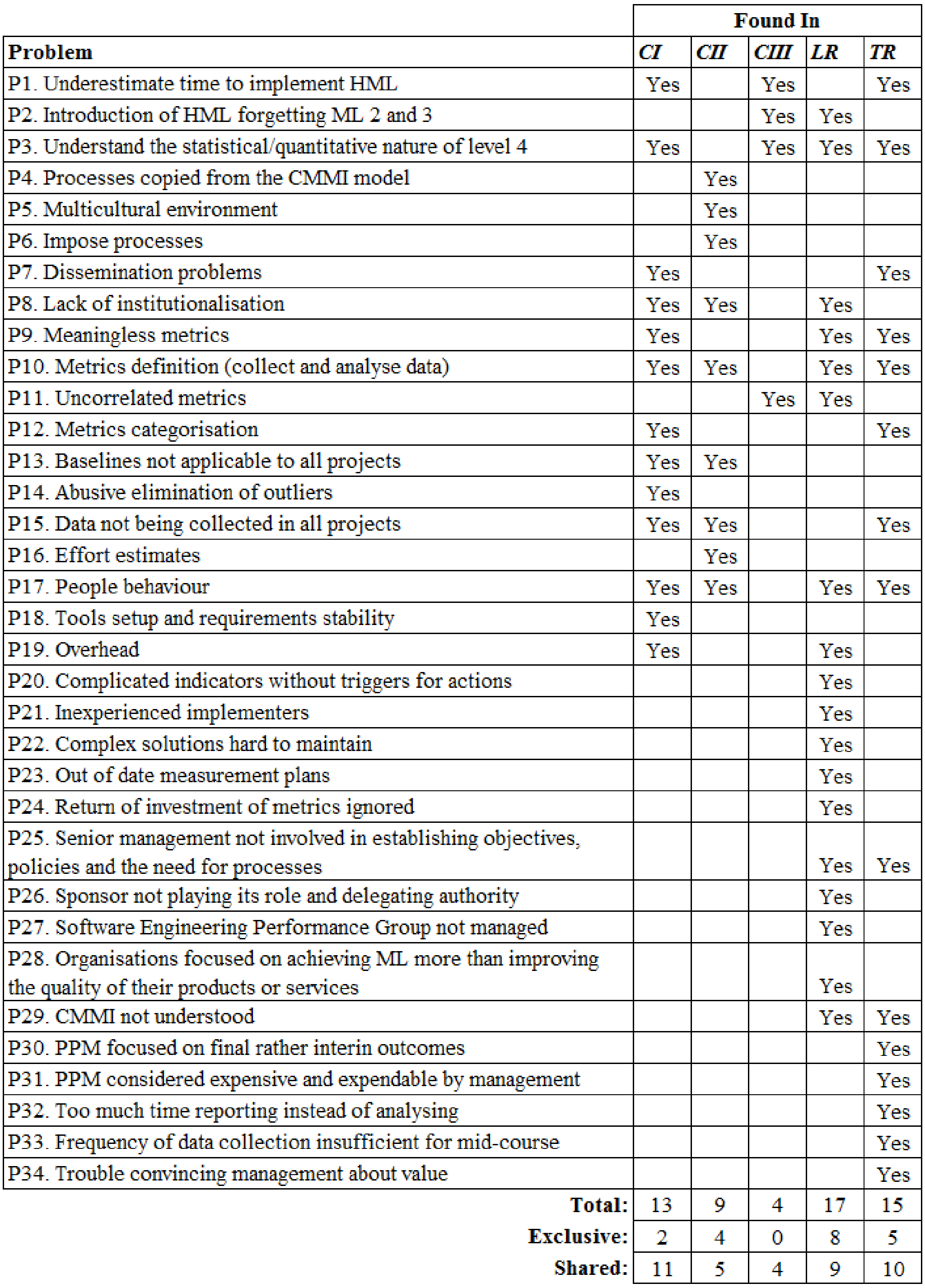
In a follow up of CII we found that these recommendations were later applied to support the definition of their specific metrics and implemented an estimation tool.

The demands of MLs 2 and 3 should prepare organisations to adequately use measurement at higher levels, by monitoring appropriate metrics. Nonetheless, some of the problems identified reflect a poor implementation of the MA PA, affecting the organisations results. Such problems become evident when implementing ML 4 because the correlation between variables and problems in the collected data affect PPM and PPB. Besides, SCAMPI cannot appraise the entire organisation and does not analyse performance measures – if it did, it would become even more expensive. Hence, CMMI rating *per se* is not a guarantee of achieving expected performance results, and organisations need to be aware that there are different methods that can be used in its implementation. Nevertheless, if some recommendations such as the ones we proposed in 4.4.2 CMMI Implementation: Problems and Recommendations are followed, CMMI implementation can be easier, and the problems discussed before can be avoided. The checklist in Table 4.1 should be used by organisations implementing CMMI to guide them in the sequence of what should be done to implement CMMI, and help them focus on the model as a whole, not just on a single target level to be achieved. It includes the problems that organisations should be aware of in order to avoid them. Most of these recommendations coincide with solutions that were used by organisations which were studied to overcome their problems, and hence validated by them.

5.1.3 Problems Analysis and Limits to Generalisation

In Figure 5.7 we signal where the problems we found occurred with "Yes". 59% of the problems occurred either in one of the organisations of our case studies, were mentioned in the literature (LR) or were found to be happening in the organisations of the TR. The number of problems we detected in each organisation increased with the depth and insight provided by a more complete design of the case study. Nevertheless, we found two groups of problems common to two different groups of two organisations.

Several problems found in CI were also detected in CII. Four of them are related to metrics definition and usage and the other two are related to institutionalisation and people behaviour, respectively. Another two problems found in CI also occurred in CIII, both of them related to assuring entry conditions. CII was just a business unit of CIIG, and was rated ML5 for a long time, so we cannot verify if they faced similar entry conditions problems. However, we realised that the metrics problems found could be due to CII lack of understanding of the requirements for HML and the statistical nature of ML4. We cannot even conclude that CIII did not face the metrics problems, because we did not analyse their PPM, PPB, metrics definitions and usage in person.



	Found In				
Problem	***CI***	***CII***	***CIII***	***LR***	***TR***
P1. Underestimate time to implement HML	Yes		Yes		Yes
P2. Introduction of HML forgetting ML 2 and 3			Yes	Yes	
P3. Understand the statistical/quantitative nature of level 4	Yes		Yes	Yes	Yes
P4. Processes copied from the CMMI model		Yes			
P5. Multicultural environment		Yes			
P6. Impose processes		Yes			
P7. Dissemination problems	Yes				Yes
P8. Lack of institutionalisation	Yes	Yes		Yes	
P9. Meaningless metrics	Yes			Yes	Yes
P10. Metrics definition (collect and analyse data)	Yes	Yes		Yes	Yes
P11. Uncorrelated metrics			Yes	Yes	
P12. Metrics categorisation	Yes				Yes
P13. Baselines not applicable to all projects	Yes	Yes			
P14. Abusive elimination of outliers	Yes				
P15. Data not being collected in all projects	Yes	Yes			Yes
P16. Effort estimates		Yes			
P17. People behaviour	Yes	Yes		Yes	Yes
P18. Tools setup and requirements stability	Yes				
P19. Overhead	Yes			Yes	
P20. Complicated indicators without triggers for actions				Yes	
P21. Inexperienced implementers				Yes	
P22. Complex solutions hard to maintain				Yes	
P23. Out of date measurement plans				Yes	
P24. Return of investment of metrics ignored				Yes	
P25. Senior management not involved in establishing objectives, policies and the need for processes				Yes	Yes
P26. Sponsor not playing its role and delegating authority				Yes	
P27. Software Engineering Performance Group not managed				Yes	
P28. Organisations focused on achieving ML more than improving the quality of their products or services				Yes	
P29. CMMI not understood				Yes	Yes
P30. PPM focused on final rather interin outcomes					Yes
P31. PPM considered expensive and expendable by management					Yes
P32. Too much time reporting instead of analysing					Yes
P33. Frequency of data collection insufficient for mid-course					Yes
P34. Trouble convincing management about value					Yes
Total:	13	9	4	17	15
Exclusive:	2	4	0	8	5
Shared:	11	5	4	9	10

Figure 5.7: Problems found in the case study organisations (CI, CII and CIII), the organisations surveyed by the SEI (TR) and the literature review (LR).

47% of problems that we found in the literature were also detected in CI, CI and CIII. 53% of problems found in the organisations surveyed by the SEI were common to the ones found in our case study organisations, 27% of which were also found in the literature review. We detected part of the problems found in LR and TR (3.1.3 Problems in Process Improvements, Metrics Programs and CMMI) and additional ones:

- Processes copied from the model (P4);
- Ignored multicultural environment (P5);
- Imposed processes (P6);
- Baselines not applicable to all projects (P13);
- Abusive elimination of outliers (P14);
- Incomplete base for effort estimate methods in use (P16);
- Difficulties in giving the tools time to become stable (tools setup) (P18);
- No baseline reset or model recalibration after tools' requirements changes (P18).

Even if there are limits to the generalisation of our results, the percentage of problems shared in more than one organisation/source indicates that they can occur when implementing HML, so organisations should be aware of them.

Due to access limitations the three case studies had a different design, so they cannot be considered multiple-case studies (Yin, 2009). Only part of the design of CI was repeated in CII, and in CIII we only interviewed a consultant involved in the appraisal. We can classify it a semi-multiple case study. In CI and CII we used multiple sources of information for confirming, and thus have more confidence in the results. However, in CIII we could not assume this. In all cases we had our results reviewed by key informants. To ensure internal validity we did pattern matching by classifying information and aggregating it under each category; built explanations and addressed rival explanations. External validity was partially tested by replicating part of the design used in CI in CII, and conducting the SDA. Nonetheless, for each case study we used theory.

Regarding limits to generalisation, we only analysed three cases but some of the problems that we identified were also found in the literature and TR. Subsequently, we consider that these problems can be common to other organisations implementing CMMI, measurement programs or doing software process improvements.

The first literature review about the problems found in CMMI and metrics programs, and the CI case study were, taken together, the stepping stone for designing the Framework and building the theory. Over the years, the recommendations and problems evolved, as more case studies were conducted (CII and CIII), and the further analysis of the SEI surveys data, contributed not only to confirm the relevance of the problems we first identified, but also to complement and prove the usefulness and value of the EQualPI's CMMI Implementation module. The case studies not only showed that some organisations may find it complex to implement CMMI HML, but also showed some problems and limitations that result from poorly implemented MA, and PPM and

PPB. With this research, we complete the answer to research question **RQ3** - *What additional recommendations can we provide to organisations to help them avoid problems when implementing CMMI?*

5.1.4 CMMI Implementation Discussion

From the organisations outcomes the CMMI model does not always deliver the expected performance results (Herbsleb and Goldenson, 1996; Charette et al., 2004; Schaeffer, 2004; Leeson, 2009; Bollinger and McGowan, 2009; Schreb, 2010) and we realised that higher maturity, that is to say, just the fact that an organisation was rated at a HML, does not necessarily mean having better quality of their outcomes – there may simply be instances of bad implementations. In EQualPI, more than the maturity of organisation we put emphasis on the quality of the outcome. In our research we could confirm that CMMI problems arise when:

- Its use is not aligned with the organisation reality, culture and business goals;
- Processes description is different from how they are actually used and there is not consistency in usage by different people;
- Interpreting processes results and acting on the current status is done while ignoring context, skewing the performance results;
- Aggregation of data is ignored, inadequate or abusive (all contexts have been bundled in the same category, once again ignoring context), therefore PPM and PPB are not faithful to the organisation's processes and do not represent all its projects;
- When imposing new practices, others that were beneficial are abandoned.

One of the reported problems in the literature and found in the organisations of our case study is underestimating the time to implement CMMI, particularly HML. Looking at the SEI reported median times to move from one ML to another (see Figure 3.3), which used to be done semiannually, we can also note that the difference from moving from ML 3 to 4 is low when compared with the time to move from ML 3 to 5. There are even semesters showing that the move to ML 5 took less time than the move to level 4. Surely the differences between semesters depend on the maturity that the organisations being appraised already have, nonetheless the median time in half of the represented semesters is lower when moving to ML 5. When taking less time to move from ML3 to 4 the difference varies between 0,5 to 4,5 months. We consider that when organisations endeavour an SPI to move from ML3 to HML, if the cost is not significantly different and the time is so close, they should make the move straight from ML 3 to 5 and fully benefit of the improvements introduced at level 4.

In EQualPI, even if the entire organisation is not being appraised the projects that are using it need to have data and processes that make sense. The baselines must serve their projects and when enlarging the scope to a BU or an entire organisation the baselines need to be adapted, the different levels of granularity must be defined with adequate variables and models introduced that give the information that is needed at the considered management level. That is not much different from the principles of CMMI but the quality of the outcome must be measurable so that, when using the process, its benefits and quality of implementation are reflected in the quality of the outcome.

If the CMMI implementation reflects ML 4 and 5 the measurement protocols are rigorous, and that is reflected in consistent data collection, analysis and interpretation; projects are using the same and adequate information; and the organisation also normalises and aggregates data such that they are available and useful at different management levels. We are aware that aggregation is not a mere sum of results of the lower levels indicators, there may be different indicators relevant at lower levels that are not needed at upper ones, other indicators may just be base for upper levels indicators, and even the thresholds established and target values of the indicators may vary from project to project, department to department, and may have more aggressive values at lower levels to ensure the desired organisation's target goals are achieved.

5.2 Requirements Process Improvement

In one of the case studies the organisation revealed difficulties in applying the existing defects classification scheme to requirements defects, which led us to demonstrate EQualPI's Process Improvement steps that are part of the Procedures Package in improving the classification of requirements defects. This section refers to the validation of part of the steps defined in 4.4.3 Process Improvements. When focusing the improvement on the classification of requirements defects our intention is to increase awareness of those defects, as their impact in later phases of the project can be high, and the cost to fix them also increases. Ideally, with this improvement, organisations will pay more attention to those defects and prevent them, thus improving their capability of detecting them in the requirements phase, lowering the quantity of those defects in later phases, and benefiting from correcting them faster than if they had to be corrected in posterior phases of the project.

To validate EQualPI's Process Improvements steps we conducted a literature review (see 3.3 Defect Classification Taxonomies) to define our process improvement and performed a field experiment. Other researchers used different taxonomies to classify requirements' defects but, at the time we performed this research, to the best of our knowledge, none of them tested the quality properties of the defect classifiers list (Lopes Margarido, 2010). While developing the list of classifiers through the analysis of other authors results we also followed their recommendations. We performed a quasi-experiment with two groups of students with knowledge of requirements engineering to demonstrate that the list of classifiers had all the properties of a good classification scheme. We were able to apply EQualPI's Process Improvement **Steps 1 to 7**. Even though we could not validate **Step 8 - Progressively deploy and control** the output of our improvement was adopted by an organisation, lending recognition to its value.

Step 1 - Identify a need and characterise the current process

We identified the need to improve the requirements review process for the afore mentioned reasons. Chen and Huang (2009) analysed the impact of software development defects on software maintainability, and concluded that several documentation and requirements problems are amongst the top 10 higher-severity problems (see Table 3.6). In the same year, Hamill and Katerina (2009) showed that requirements defects are amongst the most common types of defects in software development and that the major sources of failures are defects in requirements (32.65%) and code (32.58%) (Figure 5.8). Therefore, it is crucial to prevent the propagation of requirements defects to posterior development phases.

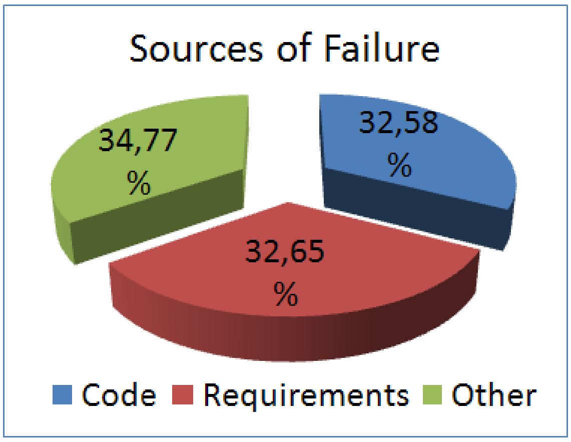


Figure 5.8: Major sources of software failures, based on Hamill and Katerina (2009).

There can be several root causes for such high number of defects being introduced in the requirements phase that can go from customer involvement, their understanding of the technology and the novelty of the need; to the way requirements are documented, namely without following standards, being incomplete, not fulfilling the needs; or in the way the requirements are reviewed before starting the development, missing important defects, without a review process or not involving the right stakeholders and experts.

Step 2 - Identify and define improvement

We identified that the limitation of not having a defects classification specific for requirements defects made analysis more complex in terms of being able to know the most common types, and for those, find solutions to their root cause. Therefore, to reduce the number of defects transferring to posterior phases, our process improvement consisted of assembling a classification scheme specific to requirements defects. Card (1998) stated that "Classifying or grouping problems helps to identify clusters in which systematic errors are likely to be found." Hence, it is important to have an adequate taxonomy to classify requirements defects, in support of the following goals:

1. Identify types of defects that are more frequent or have a higher cost impact;
2. Analyse the root cause of requirements defects;

3. Prepare requirements reviews checklists;

4. Reduce risks associated with common problems in the requirements management process, such as bad communication, incomplete requirements, and final acceptance difficulties.

ODC is frequently used by practitioners, but it is more appropriate for classifying code defects than defects in the requirements specifications (Henningsson and Wohlin, 2004; Freimut et al., 2005). There are several classifications identified in the literature, but none of them is indicated as being the most fitting for the classification of requirements defects. In our research we did a literature review (documented in section 3.3 Defect Classification Taxonomies) to define the improvement, by assembling and proposing values for the attribute type of defect in the case of requirements using the recommendations of Freimut et al. (2005).

In the context of this process improvement a **defect** is a fault, as defined by IEEE Std 610:1990, extended to include all the software development artefacts (code, documentation, requirements, etc.). A defect is a problem that occurs in an artefact and may lead to a **failure**. We consider the requirements review as an inspection method.

Step 3 - Determine selection criteria and select improvement methods accordingly

If we focused on the improvement of defects classification in general we would want to assemble a scheme of good quality and ensure that people would be able to apply it consistently. That is to say, different people would be classifying the same defect with the same classifier. Freimut et al. (2005), indicate the quality properties of a good classification scheme:

1. Clearly and meaningfully define the attributes of the classification scheme;

2. Clearly define the values of the attributes;

3. Ensure it is complete (every defect is classifiable by using the scheme);

4. Guarantee that it contains a small number of attribute values - the authors recommend 5 to 9 attributes, since this is the number of items that human short-memory can retain (Chillarege et al., 1992; Miller, 1956);

5. Aggregate attribute values, to reduce ambiguity (Bell and Thayer, 1976), whenever they are less significant, that is, when they rarely occur, and detailed categories may be aggregated into a single one. For the attribute "type of defect" we consider that it is important that the values are unambiguous, that is, only one value is applicable to one defect.

We compiled all defects classifiers we found in the literature review in Table 3.9. We wanted to reduce it to a set useful to classify requirements defects, so we removed classifiers following the criteria:

- Not applicable to requirements phase;

- Inadequate to review a document;

- Vague and generic;
- Over-detailed;
- Duplicated, or classifiers with the same meaning.

The following classifiers were excluded for the reasons listed next.

1. Important only for change management:

Not in current baseline, *New and Changed Requirement*, and *Not Traceable*.

2. Too vague, given the intention of having a complete and clearly defined list of values:

General, *Other* and *Inadequate*.

3. Subsumed by another, in this case **Inconsistent**:

Incompatible.

4. Too generic, given the existence of separate, more specific, classifiers:

Incorrect or Extra Functionality.

5. Over-detailed, given the existence of the more generic classifiers **Missing/Omission**, **Incorrect** and **Inconsistent**, and the intention of keeping a small number of attribute values:

Classifiers 19 to 33 and 35 in Figure 3.9 detailing what is missing, incorrect or inconsistent (the details can be given in the defect description).

6. With overlapping meanings, and small frequencies in some cases, that were aggregated into a single one, to avoid ambiguity:

- *Missing/Omission* and *Incomplete* → **Missing or Incomplete**;
- *Over-specification*, *Out of scope*, *Intentional Deviation* and *Extraneous Information* → **Not Relevant or Extraneous**;
- *Unclear* and *Ambiguity* → **Ambiguous or Unclear**;
- *Infeasible*, *Unachievable*, *Non Verifiable* and *Untestable/Non Verifiable* → **Infeasible or Non-verifiable**.

Finally, some classifiers were slightly renamed to aid reliable use and common understanding. The resulting 9 values for the type of defect attribute, with definitions and examples, are listed in Table 5.4. We tried to give a clear and meaningful definition for each value.

Table 5.4: Classification of type of defect for requirements (final version) (Lopes Margarido et al., 2011a).

Classifier	Definition	Example
Missing or Incomplete	The requirement is not present in the requirements document. Information relevant to the requirement is missing, so the requirement is incomplete. If a word is missing without affecting the meaning of the requirement the defect shall be classified as a typo.	"The system will allow authentication of authorised users." The way to access the system is not detailed. Is it by using a login and corresponding password? Using a card? And what happens when a non-authorised user tries to access it? If the requirement includes the expression *To be Defined* (TBD) is incomplete.
Incorrect Information	The information contained in the requirement is incorrect or false, excluding typographical/grammatical errors or missing words. The requirement is in conflict with preceding documents.	Stating that "The Value Added Tax is 23%" when the correct value is 12%.
Inconsistent	The requirement or the information contained in the requirement is inconsistent with the overall document or in conflict with another requirement that is correctly specified.	One requirement states that "all lights shall be green" while another states "all lights shall be blue" (IEEE Std 830-1998); one of the requirements is inconsistent with the other.
Ambiguous or Unclear	The requirement contains information or vocabulary that can have more than one interpretation. The information in the requirement is subjective. The requirement specification is difficult to read and understand. The meaning of a statement is not clear.	The requirement "An operator shall not have to wait for the transaction to complete." is ambiguous, depends on each person's interpretation. To be correctly specified it should be, e.g., "95% of the transactions shall be processed in less than 1 second." (IEEE Std 830-1998).
Misplaced	The requirement is misplaced either in the section of the requirements specification document or in the functionalities, packages or system it is referring to.	Include a requirement about the server application in the section that refers to the web-client application.
Infeasible or Non-verifiable	The requirement is not implementable, e.g., due to technology limitations. The requirement implementation can not be verified in a code inspection, testing or using other verification method. If the requirement is non-verifiable due to ambiguity, incorrectness or missing information, use the corresponding classifier instead.	"The service users will be admitted in the room by a teleportation system." The teleportation technology has not sufficiently evolved to allow the implementation of such requirement. "The message sent to the space for potential extraterrestrial beings should be readable for at least 1000 years."
Redundant or Duplicate	The requirement is a duplicate of another or part of the information it contains is already present in the document, becoming redundant.	The same requirement appears more than once in the requirements specification document, or the same information is repeated.
Typo or Formatting	Orthographic, semantic, grammatical error or missing word. Misspelled words due to hurry. Formatting problems can be classified in this category.	"The system reacts to the user sensibility, i.e. if the user is screaming the system stops." The word sensibility is different from sensitivity. When a picture is out of the print area.
Not relevant or Extraneous	The requirement or part of its specification is out of the scope of the project, does not concern the project or refers to information of the detailed design. The requirement has unnecessary information.	If the customer is expecting a truck then the requirement stating "The vehicle is cabriolet." is out of the scope of the project. A requirement that should have been removed but is still in the document.

5.2.1 Experiments with Students

Step 4 - Set improvement goals and how to validate them

We did not explicitly set an improvement goal in terms of what would be the expected number of defects reduction but we still set the goal that the classification of the same defect to be done unambiguously; one defect would have a single possible classification. The manner of validation of the improvement would be by conducting a quasi-experiment with students, to validate the quality properties of the proposal, measuring the level of agreement of individuals when classifying the same defect.

The validation was done by conducting an experiment of defects requirements classification using the classification list we developed. The variables and statistical tests to use were:

- Misclassification - percentage of students that used a classifier different from the one we were expecting (equation 4.4);
- Divergence - percentage of students that did not classify the defects as the majority (equation 4.2);
- Fleiss' Kappa - to measure the level of agreement between subjects to classify the same defect not by chance;
- Cochran Q - to understand if the difficulty to classify each one of the defect unanimously was the same for the different defects.

The hypotheses to test with the Cochran Q are:

H_0 - The proportion of students classifying the defects as the majority is the same for the different defects.

H_1 - The proportion of students classifying the defects as the majority is the same for the different defects.

Step 5 - Pilot the process improvement

We did a first pilot of the improvement and with the lessons learnt in the first group we improved/refined the classification list and did a pilot with a second group. We conducted two experiments with different groups of people and similar classifiers. The final list (Table 5.4) used in the second group had more detail in the values, definitions and examples. The first group was composed of master graduate students that had learnt how to develop an SRS document, and were familiar with inspections and defect classifications. The second group was composed of third year undergraduate students who were familiar with SRS documents, inspections and defect classifications. We provided to each group with the same SRS and list of its defects. The subjects should register the type of defect in a form that included: the defects to classify, and distinct fields for the classifier, doubts between classifiers or to a new classifier and corresponding definition. The classification of the defects would indicate if the classifiers were ambiguous (one defect with different classifiers), meaningless (incorrectly classified) or incomplete (new classifier suggested).

Step 6 - Analyse pilot results

In Table 5.5 we summarise the information about the two groups of students and the conditions of each experiment. We recognise that they were not ideal due to the levels of noise and anxiety of both situations. The number of students of the second group was considerably lower than the first one, which was composed of 19 students. Many of the students of the second group have left the room without participating in the experiment, only 6 of them participated and one only classified 3 defects, therefore we had to exclude the subject and just consider the answers of 5 subjects.

Table 5.5: Experiment conditions of each group.

Experiment Information	**Group 1**	**Group 2**
Education	Graduates	Undergraduates
Experience	Developing an SRS document	Learnt the theory
Subjects	19	5
Introduction	We did a presentation of the scope and instructions about the experiment	The teacher gave them the introduction
When	During a class to clarify doubts about the final project	Presented before the exam to be performed after the exam
Preparation time	3 minutes to read the classifiers list (no individual record)	Average of 3 minutes to read (2 of them skipped the reading and were the ones spending more time doing the classification)
Experiment duration	40 minutes to do the classification (no individual record)	13 minutes on average doing the classification

We analysed the first pilot results and did not find the degree of consensus in the different subjects classification of the same defect that we expected. For that reason we analysed the results to identify how to improve the classifiers list and used the improved version in a second pilot. The results of the experiments are summarised in Tables 5.6 and 5.7, respectively. We analysed the degree of Misclassification (designated as missed) of each defect through the True Positives (TP) and False Positives (FP) per defect and globally. Misclassification represents the classification error compared to the classifier that we expected the students to use for each defect. We also determined the number of students answering as the majority or choosing any other classifier, to determine the divergence between their classifications of the same defect and globally. The results show that the percentage of misclassification is higher than the divergence of classifications (around 10% in the first experiment and 14% in the second). We noticed that in the first experiment no defect was unanimously classified and in the second 6 were, representing ~21%. The second group had a very small improvement in the percentage of divergent classifications, when comparing with the first group (approximately 1%), the misclassification percentage was higher (around 3%) and there was one case where each subject used a different classifier. We cannot be sure whether the number of expected students had actually participated in the second experiment these results would be better or not.

Table 5.6: Experiment results of the first group.

Def.	**Expected**	**Most Used**	**TP**	**FP**	**Missed**	**Majority**	**Other**	**Divergence**
1	Inconsistent	Inconsistent	13	6	31,58%	13	6	31,58%
2	Typo	Typo	17	2	10,53%	17	2	10,53%
3	Missing or Incomplete	Missing or Incomplete	18	1	5,26%	18	1	5,26%
4	Missing or Incomplete	Missing or Incomplete	9	10	52,63%	9	10	52,63%
5	Typo	Typo	18	1	5,26%	18	1	5,26%
6	Infeasible or Non-verifiable	Infeasible or Non-verifiable	8	11	57,89%	8	11	57,89%
7	Ambiguous or Unclear	Ambiguous or Unclear	17	2	10,53%	17	2	10,53%
8	Missing or Incomplete	Missing or Incomplete	9	10	52,63%	9	10	52,63%
9	Infeasible or Non-verifiable	Infeasible or Non-verifiable	15	4	21,05%	15	4	21,05%
10	Typo	Redundant	3	16	84,21%	8	11	57,89%
11	Typo	Typo	18	1	5,26%	18	1	5,26%
12	Not relevant	Not relevant	17	2	10,53%	17	2	10,53%
13	Incorrect	Typo	2	17	89,47%	15	4	21,05%
14	Missing or Incomplete	Missing or Incomplete	15	4	21,05%	15	4	21,05%
15	Typo	Typo	11	8	42,11%	11	8	42,11%
16	Typo	Typo	10	9	47,37%	10	9	47,37%
17	Inconsistent	Inconsistent	6	13	68,42%	6	13	68,42%
18	Missing or Incomplete	Missing or Incomplete	17	2	10,53%	17	2	10,53%
19	Missing or Incomplete	Missing or Incomplete	14	5	26,32%	14	5	26,32%
20	Missing or Incomplete	Missing or Incomplete	12	7	36,84%	12	7	36,84%
21	Inconsistent	Inconsistent	8	11	57,89%	8	11	57,89%
22	Incorrect	Incorrect	18	1	5,26%	18	1	5,26%
23	Missing or Incomplete	Missing or Incomplete	7	12	63,16%	8	11	57,89%
24	Missing or Incomplete	Ambiguous or Unclear	5	14	73,68%	12	7	36,84%
25	Missing or Incomplete	Missing or Incomplete	18	1	5,26%	18	1	5,26%
26	Missing or Incomplete	Missing or Incomplete	15	4	21,05%	15	4	21,05%
27	Inconsistent	Missing	2	17	89,47%	16	3	15,79%
28	Inconsistent	Inconsistent	10	9	47,37%	10	9	47,37%
29	Inconsistent	Missing	1	18	100,00%	17	2	10,53%
		Total	333	218	39,56%	389	162	29,40%

Table 5.7: Experiment results of the second group.

Def.	**Expected**	**Most Used**	**TP**	**FP**	**Missed**	**Majority**	**Other**	**Divergence**
1	Inconsistent	Inconsistent	4	1	20%	4	1	20%
2	Typo	Typo	4	1	20%	4	1	20%
3	Missing or Incomplete	Missing or Incomplete	5	0	0%	5	0	0%
4	Missing or Incomplete	Missing or Incomplete	3	2	40%	3	2	40%
5	Typo	Typo	4	1	20%	4	1	20%
6	Infeasible or Non-verifiable	Infeasible or Non-verifiable	2	3	60%	2	3	60%
7	Ambiguous or Unclear	Ambiguous or Unclear	5	0	0%	5	0	0%
8	Missing or Incomplete	Missing or Incomplete	3	2	40%	3	2	40%
9	Infeasible or Non-verifiable	Infeasible or Non-verifiable	5	0	0%	5	0	0%
10	Typo	Redundant	1	4	80%	4	1	20%
11	Typo	Typo	5	0	0%	5	0	0%
12	Not relevant	Not relevant	3	2	40%	3	2	40%
13	Incorrect	Typo/Incorrect	2	3	60%	2	3	60%
14	Missing or Incomplete	Missing or Incomplete	4	1	20%	4	1	20%
15	Typo	Missing or Incomplete	2	3	60%	3	2	40%
16	Typo	Incorrect	2	3	60%	3	2	40%
17	Inconsistent	Another each	1	4	80%	0	5	100%
18	Missing or Incomplete	Missing or Incomplete	3	2	40%	3	2	40%
19	Missing or Incomplete	Missing = Redundant	2	3	60%	2	3	60%
20	Missing or Incomplete	Missing = Redundant	2	3	60%	2	3	60%
21	Inconsistent	Incorrect	2	3	60%	3	2	40%
22	Incorrect	Incorrect	5	0	0%	5	0	0%
23	Missing or Incomplete	Missing or Incomplete	3	1	25%	3	1	25%
24	Missing or Incomplete	Ambiguous or Unclear	1	4	80%	4	1	20%
25	Missing or Incomplete	Missing or Incomplete	5	0	0%	5	0	0%
26	Missing or Incomplete	Missing or Incomplete	4	1	20%	4	1	20%
27	Inconsistent	Missing or Incomplete	0	5	100%	5	0	0%
28	Inconsistent	Missing or Incomplete	1	4	80%	3	2	40%
29	Inconsistent	Missing or Incomplete	0	5	100%	5	0	0%
		Total	83	61	42,36%	103	41	28,47%

The PI we used to evaluate the process of classifying defects is the *degree of agreement* or *divergence* of classifications by different subjects. Our results showed that the degree of agreement of the subjects, given by the Fleiss' Kappa measure, was **moderate** in both experiments (0.46 in the first experiment and 0.44 in the second) (Landis and Koch, 1977). We also did a Cochran test to verify if there was a statistical difference between the difficulty of classifying the defects with the same classifier. Since the test is binomial, we considered that when the subjects chose the most used classifier they answered as the majority (1) and when they used any other classifier, they chose other (0). The significance value indicates that the number of subjects using the same classifier differs from defect to defect (0,000 in both groups). The p-value <= 0,05, indicates that we can reject H_0, a result that is coherent with the moderate degree of agreement between subjects and the ~30% of divergence of answers observed in both experiments. These observations induce us to consider that certain defects will be differently classified, for their own characteristics. Using the same transformation of data we carried out a McNemar test to have a simpler way of seeing whether the experiments had similar results, or there was an improvement in fact. The percentages of subjects classifying as the majority or using other classifier were similar in both experiments (see Figure 5.9).

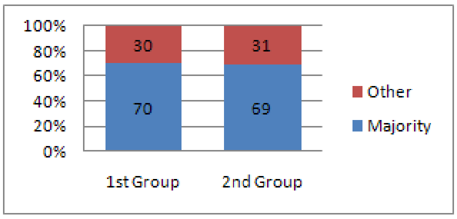


Figure 5.9: Results of the McNemar test. The experiments have approximate results.

In our opinion, the following facts may have contributed to the subjects less than expected degree of agreement:

- The experiments environment conditions were not ideal, due to the levels of anxiety and noise;
- The subjects were not the ones identifying the defects, which may increase the error of misinterpretation (and consequent misclassification) of the defects;
- The subjects were not involved in the development and did not have access to the developers of the SRS document. This is similar to the problem reported in an experiment of Walia and Carver (2007);
- Certain words in the description of defects induced the selection of the classifier named with a similar word;
- The defects are expressed in natural language, which introduces ambiguity in the classification process;

- Perhaps certain defects can have more than one valid classification.

The two experiments we did are not totally comparable: the experience of the individuals on defects classification, experience in developing requirements documents, size of the groups (19 subjects versus the 5 valid subjects) and the treatments (values of the type of defect attribute) were different. To verify an improvement in the treatment we should have involved the same group and still would need to use a different requirements document. The experiment should have been done with two groups representing the same population, one using the classification scheme, the other without it or using another scheme. Then, after improving the classifiers the experiment would have to be repeated in both groups using an SRS similar to the first document, in size, number of defects and their diversity. Furthermore, the first experiment was conducted during class and the second one after an exam, which may explain that in the second experiment we had a reduced number of participations and the speed to complete the task was higher, as the time spent in it was below the recommended. Such time recommendation was set after measuring the duration of the first experiment.

The degree of similarity of results obtained in the two experiments may reveal that it is actually hard for different people to classify some defects the same way. Even when designing the experiment the authors had to discuss some of the classifications to reach a consensus. It is also possible that the conditions of the second experiment, improvement of the list of classifiers and examples but not being able to spend the same time explaining the purpose and context of the experiment, may have contributed to deviation in the expected outcome.

Step 7 - Prepare final version

We published the final version of the requirements defects classification list. Based on the experiments conducted, we suggest some recommendations for organisations which want to use requirements defects' classifications in an effective and consensus way:

- People should be trained in the usage of the defects classification, focusing in the distinctions among classifiers, the clarification of their definitions, practical examples and exercises;
- To avoid that people apply a classifier based on its name only (often insufficient), without considering its definition, have the definition easily available, e.g., as a tool tip.

5.2.2 Adoption by an Organisation

Step 8 - Progressively deploy and control

Even though we could not fully test the process improvement and progressively deploy and control its results, the requirements defect classification we created was introduced in an organisation in 2013 as part of an improvement initiative. They considered that the use of the new requirements taxonomy successfully characterised requirements defect types when compared to using ODC where all requirements defects were being classified as documentation, therefore adopted our requirements defects types taxonomy. However, the classification was extended to have a **Not**

Applicable (N/A) and two types related with the document itself, respectively **Not Requirements - Content** and **Not Requirements - Typos or Formatting**. The classification results were used to do a Pareto chart to evaluate which requirements defect types contributed to 70% of the requirements defects: Ambiguous or Unclear (25%), Typo or Formatting (17%), Incorrect Information (16%) and Missing or Incomplete(12%) (Figure 5.10).

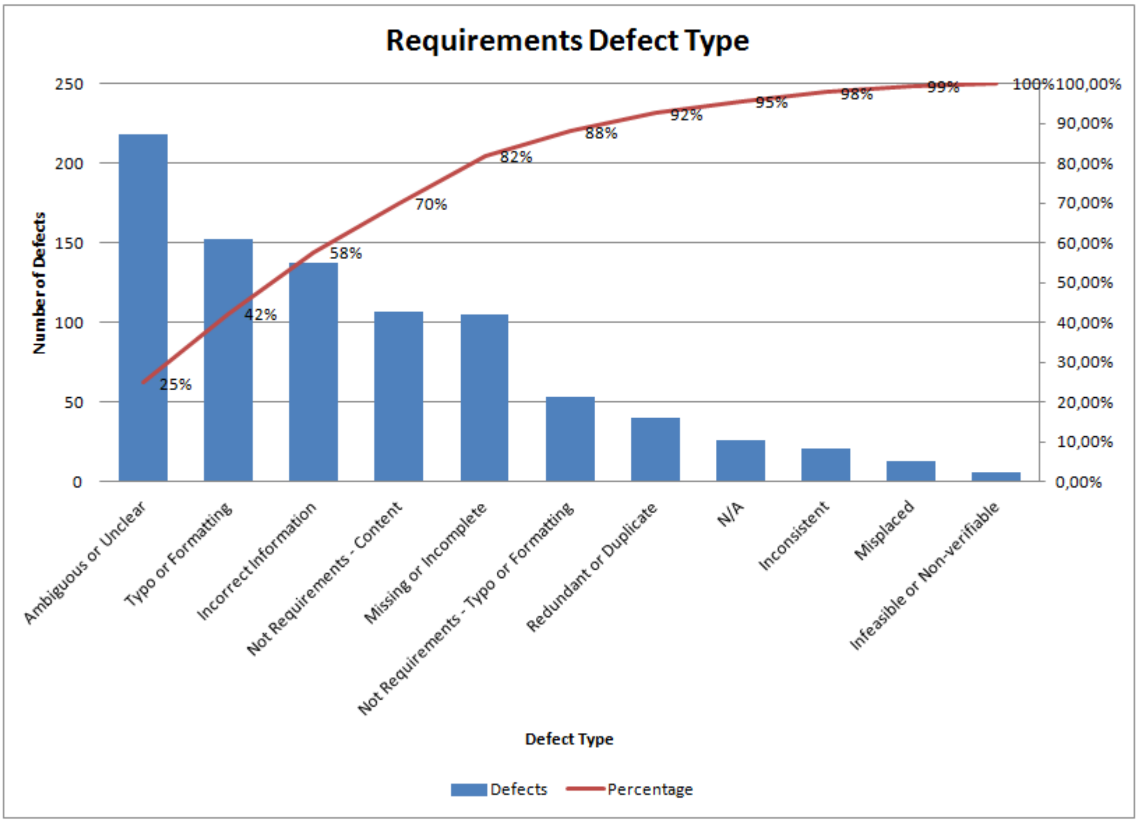


Figure 5.10: Percentage of defects found in requirements reviews by type.

The analysis results were used as inputs to a CAR project and used to define solutions to address the problems in the origin of these types of defects and prevent them in future reviews. The metric we recommended that should be monitored to improve the requirements review process was the number of requirements defects only found on posterior phases of the development cycle, although the organisation was interested in measuring other aspects of requirements reviews and introduced several other improvements that cannot be isolated in their effect from that of using the taxonomy. Nonetheless, they indicated that number of defects found in requirements was considerably higher than before introducing the taxonomy, showing an improvement.

The organisation used the defects classification scheme to support two of the goals mentioned in **Step 2** (Card, 1998):

1. Identify the requirements defects that were more frequent and had higher impact;
2. Analyse the root cause of requirements defects.

We agree with Card Card (2005) when he states that a defect taxonomy should be created in such a way that it supports the specific analysis interests of the organisation that is going to use it, namely in the implementation of defect causal analysis. In our work, based on a literature review, we assembled a classification for defect types in requirements specifications, following the recommendations given by Freimut et al. (2005). Such classification is important to support the analysis of root causes of defects and their resolution, to create checklists that improve requirements reviews and to prevent risks resulting from requirements defects. When choosing a classification for requirements' defects, organisations need to be aware of the problems of using them. People may interpret the classifiers differently, and doing retrospective analysis of defects simply based on the type of defects might be misleading. Experiments similar to the one here presented may be conducted to determine the degree of consensus amongst personnel.

In the case of our experiment we realised that in order to prevent the propagation of requirements defects to other software development phases it was important to improve requirements reviews. Besides using reviews checklists the use of an adequate classification to characterise defects found in requirements would:

- Make reviewers aware of the reason why the defect happened;
- Make requirements analysts more concious of what mistakes to avoid when eliciting requirements;
- Support the requirements analyst in the correction of the detected defects, through the additional understanding provided by the specific classification for requirements defect types;
- Support defects analysis;
- Allow to build a requirements checklist based on those defects.

With the lessons we learnt from our field experiment we also contribute to answer research question **RQ3** - *What additional recommendations can we provide to organisations to help them avoid problems when implementing CMMI?*

5.2.3 Process Improvements Procedure Analysis

Of the 8 Process Improvement Steps in EQualPI we validated 7. Of those 7, 2 were only partially validated as we could not define the metrics to monitor as indicated in **Step 2 - Identify and define improvement**. The group of subjects used to test the improvement may not be representative of the population as required in the first activity of **Step 5 - Pilot the process improvement**; it depends if they are already working as practitioners even though they may represent some at the beginning of their career and some who have already been working in the field.

To analyse the execution of the process improvement steps more objectively we mapped the activities of each step and identified them as mandatory or not. Per mandatory activity there is a maximum score of 2 when the activity is successfully performed, and 0 if it was not executed. In case of partially conducting the activity the score given is 1. We summarise the evaluation in

Table 5.8. The total score if all mandatory activities are executed is 36. Evaluating the steps we executed while conducting the experiments with the students we sum a total score of 25, having performed ~70% of the mandatory activities.

Table 5.8: Evaluation of the improvements steps we executed.

Improvement steps and respective activities	**Act.**	**Mandatory**	**Executed**	**Score**
Step 1 - Identify a need and characterise the current process	4			
Identify need		Yes	Yes	2
Root causes/Process to improve		Yes	Yes	2
Step 2 - Identify and define improvement	6			
Determine changes to implement		Yes	Yes	2
Selection criteria		Yes	Yes	2
Define metrics to monitor and set baseline		Yes	Partially	1
Step 3 - Determine selection criteria and select improvement methods accordingly	6			
Define selection criteria for the solutions		Yes	Yes	2
Brainstorm solutions		Yes	Yes	2
Select solution to implement according with the criteria		Yes	Yes	2
Step 4 - Set improvement goals and how to validate them	4			
Set target goal		Yes	Partially	1
Define methods to analyse and determine if the goal was achieved		Yes	Yes	2
Step 5 - Pilot the process improvement	2			
Conduct pilot of the improvement with a group of subjects representative of the population		Yes	Partially	1
When needed/possible have a control group not subject to the improvement		No	No	
Step 6 - Analyse pilot results	4			
Analyse pilot results		Yes	Yes	2
Determine if the goal was achieved		Yes	Yes	2
If needed repeat steps 2 to 5		No	Yes	
Step 7 - Prepare final version	2			
Publish the final version of the improvement		Yes	Yes	2
Step 8 - Progressively deploy and control	8			
Define training process		Yes	No	0
Conduct training		Yes	No	0
Gradually deploy to other subjects		Yes	No	0
Control the improvement variables to ensure there are no deviations from the goal		Yes	No	0
Total	36			25

Note: only the mandatory activities of a step receive a score: Yes = 2, Partially = 1 and No = 0. Act. indicates the maximum score of the activities of a given step.

5.2.4 Process Improvements Steps Discussion

We did not fully validate **Step 1 - Identify a need and characterise the current process** because we did not use an indicator of the current state of the process, the "*as is*", before starting the improvement. Ideally, to be able to measure the effect of the improvement we would have the number of requirements defects detected in requirements reviews and number of requirements defects detected in posterior phases, before introducing the improvement. That baseline would allow us to measure the effect of the improvement by increasing the number of defects detected in the requirements phase, and reduction of those defects in the next phases of the projects.

All other steps could be adapted to the improvement, although we would rather have validated **Step 4 - Set improvement goals and how to validate them** with more quantitative goals related to reducing the number of defects in subsequent phases of the development process, rather than just getting a good level of agreement using the classification scheme. Considering the goal of showing that a list of defects types specific for requirements defects is more appropriate than using a non-specific, the experiment should have been done with a control group that would use for example ODC to classify the defects, as indicated in **Step 5 - Pilot the process improvement**.

Regarding **Step 8 - Progressively deploy and control** we could not fully validate it, as it would require following the deployment in the organisation that adopted it and measure the effects of using it. Nonetheless, one of the quantitative goals we would set if we conducted the experiment in the organisation ourselves, to reduce the number of defects in subsequent phases, was achieved, since the organisation did find a higher number of defects in requirements reviews than before introducing the improvement, when they were using the ODC classification. Additionally, the organisation also used the defects classification to analyse and address the most common defect types and further improve the requirements process to prevent them, serving one of the purposes of implementing it: to be able to analyse and correct the causes of those defects.

The Process Improvements procedure is an application of the scientific method, focused on organisations. For that reason we consider it would be easier to follow those steps in an organisation. Nonetheless, we were able to complete almost all steps in the academic setting, showing that the process can be used, and is useful, for its purpose. While Juran's quality improvement and the Six Sigma DMAIC put emphasis in determining the root causes of defects and eliminating them, reducing "chronic waste" (Juran and Godfrey, 1998), the improvement steps in EQualPI are more in alignment with implementing an improvement not just focused on reducing defects but also on having better ways of doing the work, in line with IDEAL and PDSA, which in the end eliminates processes inefficiencies and ineffectivenesses as well.

5.3 Evaluation of the Estimation Process

One of the ways of evaluating the quality of implementation of a process using EQualPI is through modelling its performance. In this section we describe how we developed and tested a performance model to evaluate the quality of implementation of the CMMI PP SP1.4 "Estimate Effort and Cost". Considering the Effort Estimation Evaluation Model (4.6.3 Performance Indicator Models: Effort Estimation Evaluation), included in the EQualPI module **Performance Indicators Models**, it is not an effort estimator. There are many models and simulators of effort or cost estimation in the literature (3.4 Effort Estimation), and it is not our intention to develop a new one, but to develop a Framework to evaluate the quality of the CMMI practices, demonstrated on PP SP1.4. To determine the quality of implementation of the practice "Estimate Effort" we used Effort Estimation Accuracy, defined by controllable and uncontrollable factors. Similarly to the effort/cost estimation models or simulators, the data of these factors is used to determine the effort estimation accuracy.

There are several sources of data available for software engineering practitioners and researchers (Rodriguez, 2012). Some of them with open access, other are only accessible to sponsor organisations or are available for a fee. Caution must be exercised when selecting the data source to use on Software Engineering research, regardless the source, it is always appropriate to do data validity and integrity checks for noise removal. That is the case of the work done with PROMISE, which was being reviewed by Khoshgoftaar and colleagues, and Liebchen and Shepperd (2008). Another case is ISBSG, that contains metadata to describe the perceived quality, however it had not been validated to distinguish "true" from recorded data points. Moreover, the PROMISE repository only represents two segments of the industry as it mainly includes Open Source or NASA projects (Shirai et al., 2014). Besides the necessary quality analysis and having contributions of different industries, the data source must have the variables that are to be studied or variables that allow computing them. Below we list the selection criteria we used when choosing the data source and variables to build and validate the Effort Estimation Accuracy Model:

- Data source is available and we have authorisation to analyse the data.
- Data is collected while executing work in real projects.
- Repository includes data of more than one organisation and project.
- Variables have a common definition and data are consistently collected by the contributing organisations.
- Data are frequently reviewed and checked by the contributing organisations.
- Variables are collected on the execution of the estimation and software development processes.
- In case of aggregated data, the base data are available to check in order to ensure their quality and to extract additional information.

The file of compiled TSP data and corresponding database that were provided by William Nichols satisfy these criteria. Nonetheless, we still found some issues in the TSP dataset, which we addressed in the data cleaning process we describe later in this chapter (5.3.2 Data Munging). In fact, Shirai et al. (2014) identified incorrect records they found in the defects log (3,9%), missing size data (53% mainly of actual size) and defect and time logs inconsistencies (3%).

In our research we reviewed TSP to identify performance indicators that could be applied in the characterisation of Effort Estimation Accuracy. Tamura (2009), provides the arguments to follow this approach and gives three examples of PPMs built with TSP data. TSP teams collect all base measures necessary to control performance and understand the status of a project, and predict its course to completion. With other sources of information they can be used to design process performance models. Such measures, systematically collected by the team, give fine grained detail of size, defects, effort and schedule. The quality and reliability of the data is fundamental to build the PPMs needed for CMMI HML. Our research questions were the following.

Regarding the TSP estimation process:

1. What is the predictability of the estimation model?
2. What is the percentage of the actual effort that the variables of the estimation model explain?
3. Of the variables that are estimated during the estimation process, hence used in the estimation model, which ones better predict the actual effort?

Regarding the Effort Estimation Accuracy Model:

4. Is the variable Effort Estimation Accuracy useful to evaluate the quality of implementation of the estimation process?
5. Which variables better predict the accuracy of the effort estimation process?
6. How much of the Effort Estimation Accuracy can be explained by such variables?

Jørgensen (2007), showed similar concerns to the one that lead us to evaluate the quality of the estimation process rather than developing another effort estimation model, questioning the "effect of issues of system dynamics on the meaningfulness of accuracy measurement, and problems related to the outcome-focus of the measurement, i.e., the fact that we are only evaluating the outcome of an estimation process and not the estimation process itself." He also points out problems related to the definition and interpretation of the term *effort estimates*. The people giving estimates may not know how they will be interpreted, and estimates from analogy-based models may not be comparable with the ones from regression-based models. Since optimisation functions can differ some models can systematically provide higher estimates than others.

EQualPI defines that EEA can be used as a leading indicator of the quality of implementation of the effort estimation process, when defined by a set of controllable and uncontrollable factors. The controllable factors are the ones defining the process and its outputs as a choice of what is being estimated, and considered both in the estimation process and the methods to do the estimation. Besides, once the development cycle begins, other variables will also influence the actual effort, and consequently affect the EEA (recall Figure 4.21). To define EEA we have to determine all the variables that influence it. Some of these variables are controllable factors that can be influenced and changed, others are uncontrollable factors, that the team has no power to influence nor change. The following hypotheses are the ones we consider important to establish the validity of EQualPI as applied to the estimation process. The ones in bold are the ones we could test, while the others, due to insufficient or inexistent data, could not be tested.

Related to the process:

We expected that organisations maturity, indicated by their CMMI rating, or that a more mature process (having more experience in using the same estimation process), would translate to better EEA, in alignment with the principle that a process is better based on the quality of its outcome.

HA- There is a difference between organisations of different CMMI levels, organisations with higher CMMI levels have better EEA.

HAA- Organisations using CMMI have better EEA than the ones not using CMMI.

HAB- Organisations rated at HML have better EEA than organisations rated at LML.

HB- Different organisations using the same estimation methods present different EEA results.

HBB- Organisations with more experience (number of projects) using the same estimation methods have better estimation results.

We also considered that projects where the team was involved and the team members knew their individual performance would get better estimates. We could not fully test this hypothesis as we did not have the enough data points in our sample.

HC- When PSP data is used on effort estimation EEA is better than when it is not used.

HE- When relative sizes are used EEA is worse when compared with using historical data of size of similar functionalities.

Estimation variables:

We considered that projects that estimated size would get better estimation accuracy, because CMMI recommends that size is estimated and so does TSP.

HF- Estimates based on size improve the estimation accuracy.

Time to detect and fix defects should be considered in the project plan to predict the amount of necessary rework.

HG- Projects that estimate number of defects have better effort estimates.

HI- Projects that consider time to fix defects have better estimation results.

Breaking down complex products and considering all project phases needed to conduct the project, not only to develop the product but also to plan, manage and review results and status, for example, was one of the variables we considered that would improve EEA. Planning time for those phases would be as important as including design and quality phases, to improve product quality.

HH- Projects that estimate other phases based on size that are not code have better effort estimates (HLD (High Level Design), Requirements, DLD (Detailed Design)...).

Comparison:

When organisations use their own data the models reflect their reality. However, for the first cycles such data may not be available, and using benchmarks can be an alternative to base estimates on.

HJ- Projects that use historical data produce better estimates.
HK- Using benchmark data improves the estimates when no other source is available.

People experience:

Experience in a process, whichever process we consider, and in the same type of project, should be reflected on the quality of the outcome of executing the process. Therefore, we expected it to also be a variable that positively influenced the quality of the estimates.
HL- Experts' experience influences the estimation accuracy. *Assuming estimates are done by experts*
HLA- Experts' Experience in number of projects improves the accuracy of the estimates.
HLB- Experts' experience in number of years improves the accuracy of the estimates.
HM- Planning done with the elements participating in the project improves the accuracy of the effort estimates.
HMA- TSP experience of individuals improves the accuracy of the estimates.
HMB- PSP experience of individuals improves the accuracy of the estimates.
HMCA- Experience of individuals using the same technology in number of projects improves the estimates.
HMCB- Experience of individuals using the same technology in number of years improves the estimates.
HMDA- Experience of the project manager on technology in number of projects improves the estimates.
HMDB- Experience of the project manager on technology in number years improves the estimates.
HZ- Productivity improves with time when coding the same functionality. Unless the developer is already familiar with the task and technology, in which case the productivity would be more stable.

Estimation process and project phases considered:

Tasks that are kept small and sufficiently broken down give a better idea of what is needed to do the work and help stop avoid forgetting important tasks. Re-estimating at the beginning of a new cycle should improve EEA and the acquired knowledge from previous cycles can be considered.
HN- Granularity of tasks improves tasks stability and consequently effort estimation accuracy.
HO- Re-estimation at the beginning of development cycle or phase improves the effort estimation accuracy.
HP- Changes in planned project artefacts and development phases decrease the estimation accuracy.

The architecture of the product helps give engineers the bigger picture of what is being implemented, how the components integrate, how the product integrates with outside systems and how it can evolve. That clearer view helps planning of the components to develop, the shared parts that should be centralised instead of reimplemented on each component and allows better understanding of which tasks are needed to implement the product.

HQ- Designing the architecture improves the estimation accuracy.

HR- Doing detailed design improves the estimation accuracy.

HS- Traceability between requirements, architecture, design, code and tests improves the estimation accuracy.

HT- Quality phases improve the estimation accuracy.

The way the project progresses, and having the team who committed to the plan available to work, should find reflection in the accuracy of the produced estimates.

HU- Stable teams have lower effort estimation error.

HV- Team members availability to work reduces the effort estimation error.

HX- Interruptions (measured in PSP) worsen the effort estimation accuracy.

We can not forget other factors related to experience, complexity, training and knowledge of the process.

HW- Projects characteristics (complexity, novelty, project duration, temporal horizon of the estimates, team size, team experience, client knowledge, relation with client) influence the effort estimation accuracy.

HWA- Project duration increases the effort estimation error.

HWB- Project complexity increases the effort estimation error.

HWC- Solution novelty increases the effort estimation error.

HWE- Team size increases the effort estimation error.

Learning effects, proximity to the customer and its familiarity with the technology, and stakeholders involvement would also improve the estimation and development processes, consequently improving EEA.

HWF- Team experience improves the effort estimation accuracy.

HWG- Client knowledge of the technology and system improves the effort estimation accuracy.

HWH- Closeness of relations with the client improves the effort estimation accuracy.

HY- Some phases are more predictable than others (have lower estimation error).

We validated part of these hypotheses with the data we had access to, by analysing if the factors were part of the EEA model or by comparing the differences of the effort estimation accuracy between groups, distinguished by having or not having the characteristic to test, for example. We also mapped them with the results documented by other authors when testing the same or similar hypotheses (see 3.4 Effort Estimation). We consider that organisations that have these indicators are able to manage and improve their estimation process better, as they can consider those factors when estimating and will be able to act on them when they want to improve quality while controlling cost.

5.3.1 Data Extraction and Characterization

We received a list of projects, already with aggregation of plan and actual variables, where the individual effort records had already been validated by the SEI through the Benford's law (Sasao et al., 2010). We also had access to the TSP database of Excel workbooks, from which we extracted the data required in EQualPI's Data Dictionary[2], including the one needed to analyse the estimation process. We could "reverse analyse" the data table that held the information to understand how the estimation process used was related to the tasks. We extracted tasks with actual hours higher than 0, which had a size estimate. Even though this procedure introduces researcher bias, we cannot guarantee that the reason for the time spent on the task to be 0 is due to overestimation, and therefore the task was unnecessarily planned, or the workbook is incomplete because it was uploaded to the database before completing the cycle or project. Thus, we only gathered data of completed tasks.

In our data analysis we wanted to use variables based on estimated, planned and actual values of effort, time in task measured in hours, size and defects in the dimensions number of detected defects and time spent fixing them. We observed that only two workbooks had planned injected defects[3], none of them planned injected defects in documentation (in TSP the considered documents are requirements, design, detailed design) and no workbook had estimated defects removed. Consequently, we did not consider defects estimation and respective effort in the model.

We tend to consider that there are phases where defects are injected and phases where they are removed, when the quality filters are used. However, the database shows injected defects in phases that are of defect removal. This behaviour could be expected in testing phases, as it is common that solving a problem may inject a defect when developers do not consider all dependencies, but, interestingly enough, we also found defects injected in code inspections and reviews, where normally we would not expect the injection of defects. We want to alert organisations to these facts, so they consider them when planning their projects. We noticed that teams had planned the defects injected per phase but did not plan for the defects removed per phase.

5.3.2 Data Munging

The TSP Database had information of 257 workbooks, of which 234 had size estimates, while the consolidated projects list only had 114, four of which were not present in the database version we analysed. When checking the data to guarantee uniqueness of projects there is no field that allows us to group all workbooks of the same project together and the ones in the consolidated list were classified with the ID (unique identifier) of the workbook. We found several duplicates, which we removed, ending with a sample of 88 projects to analyse. Of those, six do not have any part measured in LOC, reducing the sample to 82 projects.

We removed duplicated workbooks, just keeping the most recent version with more complete information, using the following check criteria to find similar data:

- Same number of defects or consistently increasing number of defects on the same phases;

- Defects with the same descriptions;

[2] That is part of the Repository package.

[3] After cleaning duplicates there was just one project that estimated defects injected.

- Same increasing schedule, i.e. sharing start week and having increasing number of weeks, increasing actual/plan in last weeks hours, same actual/plan hours in the same completed weeks and same Plan Schedule;

- Same team members on tasks with the same descriptions.

We also noticed the existence of 352 tasks with Plan Hours equal to 0. Of those, 38 were estimated, as they had estimated hours but the number of engineers was 0, hence, when multiplied the plan hours resulted in 0. A total of 348 tasks had 0 planned hours and did not use the estimate, which could mean they were not considered when the plan was consolidated. The other tasks without estimated hours were not estimated at all. They could be considered as estimated but not executed, but being a small number and not having information to understand the reason for that to happen we removed those data points (~0,5% of the sample).

After cleaning the data we ensured that they were correctly classified, in the case of nominal variables; for that we needed to have clear factors. That was the case of the variables **Size Measure** and **Phase Name**. On both we found several values to describe the same factor, incorrectly increasing the number of factors. The Size Measure factors went from 75 to 28 levels, while the 30 levels of Phase factors were corrected and reduced to 23. In the case of the Size Measure we found several **Components** (or parts) with same ID and Size, only having size measure in some of the development phases, so we did the correction, when possible, analysing them case by case. Similarly, we corrected when possible missing values of other Phases that corresponded to parts of the same size.

5.3.3 Process Variables Definition and Data Aggregation

We defined **Process Variables** based on the TSP planning and quality plan guidelines (Humphrey, 2006), whose recommended values are presented in Table 5.9. We considered those values to define the Process Variables that are indicated and described in Table 5.10 to build the EEA model.

We defined the variables at different levels of aggregation, task, component and project, according to the scope where they are meaningful and can be defined. Not all variables are interpretable at task and component level, for example ratio variables that are based on two development phases, cannot be calculated for a single task as each one has just a phase. These variables may not be interpretable even at component level, because many components do not have all possible project phases. Hence, we built the model at the project level. The aggregation at component level was not possible by simply stating that a component or part has a unique ID as we did to define the project ID, so that after removing duplicates could be unequivocally identified. Therefore, we defined the component as the parts with same **ID** implemented in the same project, which have same **Size**, use the same **Size Measure** and are estimated at the same **Rate per Hour**. These conditions are necessary to determine the value of certain Process Variables that depend on the component total estimated, planned and actual times. Additionally, we need to consider that at

Table 5.9: TSP Planning and Quality Plan Guidelines (Humphrey, 2006) that we considered in our model.

Variable	**Guideline Value**
Requirements Inspection / Requirements	> 0.25
High Level Design Inspection / High Level Design	> 0.5
Detailed Design Review / Detailed Design	> 0.5
Detailed Design / Coding	> 1
Code Review / Coding	> 0.5
Detailed Design	22.1%
Detailed Design Review	11.1%
Detailed Design Inspection	8.8%
Coding	20.0%
Code Review	10.0%
Compiling	3.4%
Code Inspection	8.8%
Unit Test	15.8%
Rate per Hour for New or Large modifications	10 LOC/hour
Small Changes to Large Systems	5 LOC per hour
Code Reviewed per Hour	< 200 LOC/hour

task level many of the guidelines are defined as followed, 1, or not followed, 0, but when aggregated at component or project level they are determined as a percentage of tasks or components where the guideline was followed.

Table 5.10: Process Variables used to verify process compliance or determine the value of the planned metrics that define the process variable.

Variable	**Description**
Size Used	Percentage of tasks that used the size estimate to plan the hours spent in a task by multiplying the size by the estimated hours and number of engineers.
<Phase> Percentage	Considering all phases of the project, percentage of time spent on a given Phase (e.g. Requirements).
<Implementation Phase> Percentage	Considering only implementation phases, percentage of time planned to be spent in that particular phase (see implementation phases in 2.6 Effort Estimation in CMMI and TSP).
<Implementation Phase> Allocation Percentage	Considering only implementation phases, percentage of the allocation guideline recommended to be planned to spend in that particular phase.
<Defect Removal Phase> / <Defect Insertion Phase> Value	Ratio between the time planned for a reviewing or inspection phase and the corresponding defect insertion phase, e.g. between Code Inspections and Coding.
<Defect Removal Phase> / <Defect Insertion Phase> Followed	Percentage of tasks or components where the respective estimated defect insertion and removal time were planned following the guideline
Implementation Rate	Percentage of tasks or components that used the implementation rate guidelines
Review Rate	Percentage of tasks or components, with code review phases (Code Inspection or Code Review) that followed the recommended review rates.

In the case of times, ratios and rate guidelines, if a project did not follow a guideline to estimate, that may mean either that they used their TSP historical data or that such data did not exist for the particular project, and the guidelines were not suitable in that context. Consequently, the estimates may have been based on an industry benchmark or expert judgement, as found to be more suitable for the particular project/team.

5.3.4 TSP Estimation Model

During launching, TSP teams execute a series of processes that allows them to produce the project plan. Several steps are executed on the sequence of activities to produce the final plan which the team is committed to. If we do the linear regression model of the Actual Hours and Plan Hours we get how much of the TSP development process is explained by the estimation process. The planned hours are the result of estimating the hours to spend on each project phase, by aggregating the planned hours of each of the tasks to execute the project. However, not all phases have the same weight in the plan. We built regression models to know how much of the variation of Actual Hours is explained by the Plan, in order to know exactly how much of the actual hours can be explained by the components of the estimation process outputs with meaningful significance. The purpose of the models is to understand the predictability of the TSP estimates to then study and build an EEA accuracy model based on the variables used to define the plan. The ultimate goal is to allow organisations to anticipate their expected estimation accuracy and know which variables to act on in order to improve it before having a final version of the plan.

The first model was produced to understand the predictability of the estimation process, i.e. the percentage of the actual effort spent in the project that is explained by the plan. We verified that the correlation between actual hours and estimated hours was lower than the plan hours, indicating that the variable Planned Hours is the most accurate in predicting Actual Hours. The mathematical difference between estimated and planned is that the latter considers number of engineers. However, we found that was not always the case, because the planned hours are determined considering the top down and bottom up estimation necessary to define the plan. Therefore, we defined all prediction variables based on planned hours. We did linear regression using the method *Enter*, considering the cases *Listwise*, since all projects had data. The regression model of Actual Hours as a function of Planned Hours has an Adjusted R Square of 92,8% (see equation 5.2). In this case the R Square could be used, since we have just one variable; its value is 92,9%.

$$Actual = f(Plan): \quad ActualHours = \beta_0 + \beta_1 PlanHours + \varepsilon \tag{5.1}$$

$$ActualHours: \quad ActualHours = 24,692 + 1,164 PlanHours + \varepsilon \tag{5.2}$$

The variable Plan Hours results of the sum of all hours that were planned for all project tasks. We built the regression model of actual hours, as a function of the data gathered to build the plan, showing the variables that for these data explain actual hours, i.e. variables of time, estimated using TSP, and Team size. Once again we used the linear regression method *Enter*, considering the cases *Listwise*, since all projects had data. For the model we selected all variables retrieved by an automatic model and removed team size as it was not significant. Considering we are doing a multivariate analysis we consider the Adjusted R Square of the model, which is 92,8%, and its equation is expressed in 5.3. We present all the coefficients and respective significance of the Actual Hours model in Table 5.11.

$$ActualHours = f(est.param.): \quad ActualHours_1 = 63,968 + 6,348 CompileTime + 3,132 UTTime + 1,685 CodeInspTime + 1,273 DLDTime + 0,864 CodeTime + \varepsilon \tag{5.3}$$

Table 5.11: Actual Hours model coefficients, all variables are significant and the significance level of the model itself is 0,000 with adjusted R Square of 92,9%.

Variables	**Beta**	**Std. Error**	**Sig.**
Intercept	63,968	47,219	0,182
COMPILETime	6,348	2,769	0,026
UTTime	3,132	0,880	0,001
CODEINSPTime	1,685	0,582	0,006
DLDTime	1,273	0,340	0,000
CODETime	0,864	0,298	0,006

The models have very close adjusted R square, 92,9%, a significance level of 0,000 and all variables are statistically significant. The adjusted R Square indicates how much of the dependent variable is explained by the ones in the model rather than by chance. The latter model indicates which variables of the plan actually explain 92,9% of the actual hours, the ones that were actually meaningful to the execution of the project. The variables of planned time in phase that are more significant in the model are Compile and Unit Testing; a variation of one unit in one of them results in a bigger increase in the actual effort spent in the project, of 6,348 and 3,132, respectively. On the other hand, a variation of one unit of coding time has less impact in the Actual Hours, an increase of 0,864 hours. The effect of varying 1 unit of time in Code Inspections and Detailed Design increases the Actual Hours in 1,685 and 1,273, respectively. We can notice that the variables that better explain the actual hours are all part of the implementation phases. It may be because those variables have recommended percentages of time in phase in the TSP guidelines, being more common in TSP projects. Furthermore, the fact that defect removal phases are followed by additional work may be another reason why implementation phases have higher impact on the actual hours. We were expecting team size to be a significant variable in the TSP estimation model, but in this case it was not.

The TSP model is already accurate as can be seen from the adjusted R Squares of equations 5.2 and 5.3. However, there is still a deviation from the actual results that is not explained, in the case of our data this represents around 7% . Effort estimation models can be used to help estimate the needed effort to execute a project but how can we anticipate how far from the actual effort will the project plan be? The effort estimation accuracy as defined in equation 4.6 can only be calculated after the project execution. To make it a leading indicator we modelled it, using a similar process as the one above and exploring the estimation process variables that contributed in deciding the values of the tasks planned hours. The EEA model was developed to help explain the deviation and know which variables should be considered to improve it. It can be used with two purposes:

- Evaluate the quality of estimation when planning a project by predicting the deviation from the actual that will be expected;
- Support the decision makers whether to act on the expected deviation or not, by acting over the model variables to reduce the EEA.

5.3.5 Effort Estimation Accuracy Model

To build the EEA model we did multivariate analysis and had to ensure that our continuous variables were standardised. For that reason we used percentages of time in phase (in the overall project and regarding the development phases), and the ratios durations of phases as for to the definitions of the TSP guidelines themselves. We studied the effects of the independent variables individually on the dependent variable EEA (equation 4.6) and MER (equation 3.4). This approach allowed us to anticipate which predictors would be part of the model. Nonetheless, we also took into consideration that, when analysed together, the effects of the predictors are different.

Figure 5.11 presents the histogram and the descriptive statistics of each dependent variable. None of them follows the normal distribution. The EEA is slightly positively skewed, while MER is much more positively skewed. The number of projects that had negative EEA is also indicative that teams tend to overestimate rather than underestimate.

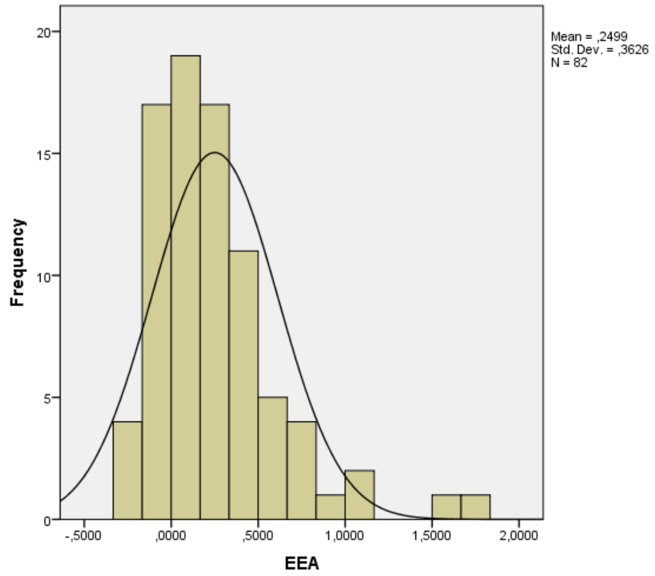
The EEA model was designed not only to evaluate the quality of implementation of the process but also to be used as a decision making tool, allowing teams/managers to decide if the plan is detailed enough to execute the project. Our requirements for the model, being aware that the adjusted R Square could be low, were the following:

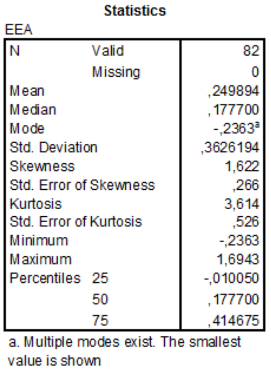
- Be a significant model (level of significance < 0.05);
- Only have significant coefficients;
- Not have collinear coefficients;
- Prevent model over-fitting by having a reasonable number of k coefficients in the model when compared to the number n of subjects.

Regarding the last rule, k should follow the equation 5.4 (Knecht, 2005; Tabachnick and Fidel, 2000):

$$NumberofCoefficients: \quad k < \frac{n}{3} \;\; or \;\; k < \frac{n}{10} \tag{5.4}$$

The number of observations in our case was 82 projects, so following the rule of thumb of keeping the number of regressors lower or equal to 8 allows us to prevent over-fitting. We decided



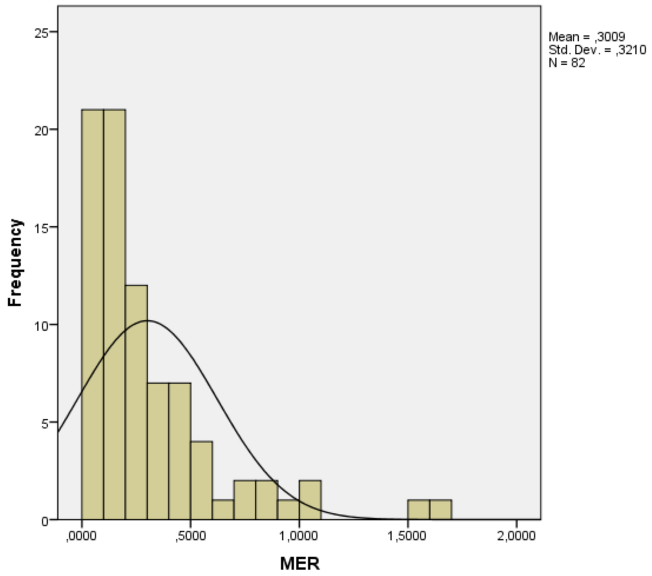


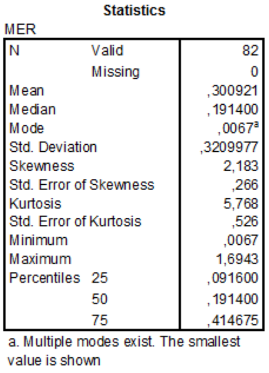
Statistics

EEA

N	Valid	82
	Missing	0
Mean		,249894
Median		,177700
Mode		-,2363a
Std. Deviation		,3626194
Skewness		1,622
Std. Error of Skewness		,266
Kurtosis		3,614
Std. Error of Kurtosis		,526
Minimum		-,2363
Maximum		1,6943
Percentiles	25	-,010050
	50	,177700
	75	,414675

a. Multiple modes exist. The smallest value is shown





Statistics

MER

N	Valid	82
	Missing	0
Mean		,300921
Median		,191400
Mode		,0067a
Std. Deviation		,3209977
Skewness		2,183
Std. Error of Skewness		,266
Kurtosis		5,768
Std. Error of Kurtosis		,526
Minimum		,0067
Maximum		1,6943
Percentiles	25	,091600
	50	,191400
	75	,414675

a. Multiple modes exist. The smallest value is shown

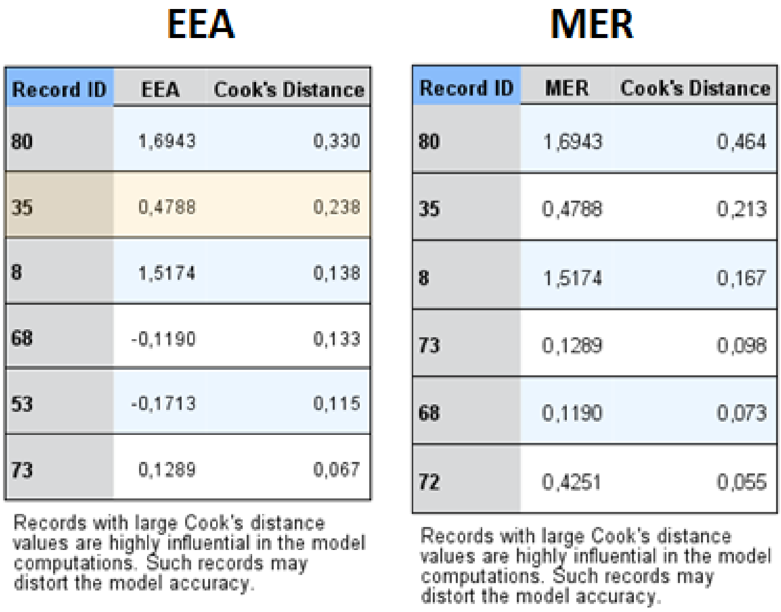
Figure 5.11: EEA and MER histograms and statistics.

Note: the upper graph and table refer to EEA and the lower to MER.

not to do any transformations to the variables to improve precision because the goal of the research was to provide additional information that can be interpreted and explained and be ready to use by practitioners.

First we tested a model considering all the guidelines variables using SPSS Automatic Linear Modelling using the *Forward Stepwise* method, using the *Information Criterion* for entry/removal, to select the variables that are significant to the model and have an effect in the dependent variable. In the automatic model generation, SPSS trims outliers, replaces missing values and transforms the variables. Therefore we used it only to guide us in the selection of the right variables. Next we did a linear regression, considering only the variables from the last step of the model, using the *Enter* regression method and *Pairwise* cases to handle missing values. We opted to use *Enter* because we wanted to check the individual contribution of the variables which, together, improve the prediction of the dependent variable and if any of them were not significant to the model we would remove them in the next execution. We selected the *Pairwise* method instead of the *Listwise* since only one project had data for all the variables. Moreover, we considered it inadequate to replace missing values by means or medians, as the missing cases could either mean that the projects did not use those phases, or the workbook in the TSP database was incomplete, e.g. corresponding to an intermediate cycle of the project. The procedure led to a statistically significant model with adjusted R Square of 0,295 and 4 coefficients, all of them significant as well. This means that the model explains 29,5% of the variability of the data other than by chance.

Even though the automatic regression model identified outliers (see Figure 5.12), we decided not to remove them since we did not have context to explain those cases. If we removed them the Adjusted R Square of the model would increase but that could also result in model overfit, given the reduced number of projects and their diversity.



EEA

Record ID	EEA	Cook's Distance
80	1,6943	0,330
35	0,4788	0,238
8	1,5174	0,138
68	-0,1190	0,133
53	-0,1713	0,115
73	0,1289	0,067

Records with large Cook's distance values are highly influential in the model computations. Such records may distort the model accuracy.

MER

Record ID	MER	Cook's Distance
80	1,6943	0,464
35	0,4788	0,213
8	1,5174	0,167
73	0,1289	0,098
68	0,1190	0,073
72	0,4251	0,055

Records with large Cook's distance values are highly influential in the model computations. Such records may distort the model accuracy.

Figure 5.12: EEA and MER models outliers.

The model for the Effort Estimation Accuracy, measured with EEA is the one on equation 5.5.

$$\begin{aligned} EffortEstimationAccuracy: \quad EEA =& \beta_0 + \beta_1 CRCODEVal + \beta_2 CMMI \\ &+ \beta_3 DLDRPerc + \beta_4 CODEINSPPercDev + \varepsilon \end{aligned} \tag{5.5}$$

Where β_0 is the intercept; CRCODREVal is the value of the ration of time spent in Code Review (CR) and implementing Code; CMMI is the maturity level; DLDRPerc is the percentage of time of the total project spent doing DLR Reviews (DLDR); and CODEINSPercDev is the percentage of implementation time spent doing Code Inspections. CMMI could also be considered a factor model that varies from 1 to 5, even though 4 is not represented in the sample data that we analysed. A change of 1 unit on each of the coefficients individually represents a change of β_n units in the Effort Estimation Accuracy. "The error term ε represents all sources of unmeasured and unmodelled random variation (Leek, 2013)" in the Effort Estimation Accuracy. With the data we used, β_n assume the values indicated on the regression model in equation 5.6.

$$\begin{aligned} EEA =& 0,251 + 0,683 \times CRCODEVal + 0,142 \times CMMI \\ & - 0,032 \times DLDRPerc - 0,019 \times CODEINSPPercDev + \varepsilon \end{aligned} \tag{5.6}$$

The EEA model has an adjusted R square of 29,5%, and can be used by practitioners to evaluate the expected accuracy of their effort estimation process, and improve it by varying the coefficients. These coefficients are the ones that explain how accurate the plan was or if it was followed or not. Knowing that a variation of 1 unit in the percentage of total project time spent reviewing the Detailed Design will cause a decrease in the EEA of around 0,032, if the variation is in the percentage of time spent in the development phases to do Code Inspections will cause a decrease of around 0,019 in EEA, the effect of varying one unit in the ratio between time spent in code reviews and time spent coding will increase the EEA in 0,683, and the level of CMMI increases the EEA in 0,142.

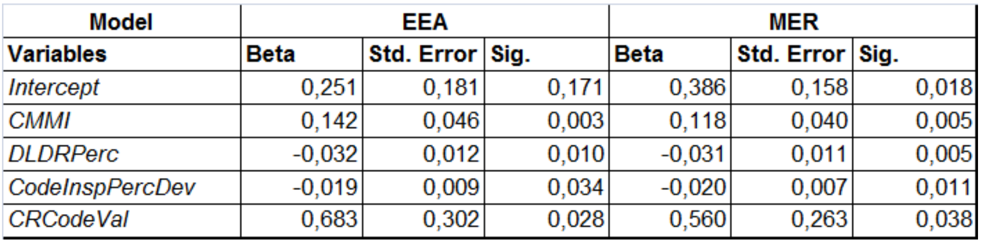
We did a regression model for MER as well, because from the indicators commonly used in Software Engineering when comparing estimation models (see 3.4 Effort Estimation), MER measures the inaccuracy relative to the estimate (Foss et al., 2003). Both dependent variables are modelled by the same coefficients but with different magnitude. The Adjusted R Square of MER is slightly higher, 31,8%. The MER model is presented in equation 5.7, followed by the equation with the coefficients values in equation 5.8.

$$\begin{aligned} MagnitudeofErrorRelative: \quad MER =& \beta_0 + \beta_1 CRCODEVal + \beta_2 CMMI \\ & + \beta_3 DLDRPerc + \beta_4 CODEINSPPercDev + \varepsilon \end{aligned} \tag{5.7}$$

$$\begin{aligned} MER =& 0,386 - 0,031 \times DLDRPerc - 0,020 \times CODEINSPPercDev + \\ & + 0,560 \times CRCODEVal + 0,118 \times CMMI + \varepsilon \end{aligned} \tag{5.8}$$

A variation of 1 unit in the percentage of total project time spent reviewing the Detailed Design will cause a decrease in the MER of around 0,031, if the variation is in the percentage of time spent in the development phases to do Code Inspections will cause a decrease of around 0,02 in MER. The effect of varying one unit in the ratio between time spent in code reviews and time spent coding will increase the MER in 0,560 and the level of CMMI increases the MER in 0,118.

In Figure 5.13 we present the coefficients data of the EEA and MER models. The standard error of the coefficients is lower on MER. All our models have less than 8 variables, respecting the rule to avoid overfit.



Model	EEA			MER		
Variables	Beta	Std. Error	Sig.	Beta	Std. Error	Sig.
Intercept	0,251	0,181	0,171	0,386	0,158	0,018
CMMI	0,142	0,046	0,003	0,118	0,040	0,005
DLDRPerc	-0,032	0,012	0,010	-0,031	0,011	0,005
CodeInspPercDev	-0,019	0,009	0,034	-0,020	0,007	0,011
CRCodeVal	0,683	0,302	0,028	0,560	0,263	0,038

Figure 5.13: EEA and MER coefficients: Beta, Standard Error and Significance

Both models are summarised in Table 5.12. The respective ANOVA can be found in Table 5.13. The MER regression model explains more variability than the EEA. We used the Durbin-Watson to test the null hypothesis, i.e. that there is no autocorrelation between the residuals. Both models present a value higher than the upper Durbin-Watson Statistic[4] for models with $K = 4$ regressors and $n = 82$ subjects (> 1,743) and the test statistic is close to, but still lower than, 2 so the residuals are not auto-correlated.

Table 5.12: EEA and MER Models summaries.

Parameter	**EEA**	**MER**
R (Person's coef.)	0,586	0,604
R Square	0,343	0,364
Adjusted R Sq.	0,295	0,318
Est. Std. Error	0,304	0,265
Durbin-Watson	1,880	1,970

Table 5.13: EEA and MER ANOVA

	Parameter	**EEA**	**MER**
Regression	Sum of Squares	2,663	2,216
	Degrees of Freedom	4	4
	Mean Square	0,666	0,554
	F Statistic	7,185	7,885
	Significance	0,000	0,000
Residual	Sum of Squares	5,095	3,864
	Degrees of Freedom	55	55
	Mean Square	0,093	0,070
Total	Sum of Squares	7,758	6,079
	Degrees of Freedom	59	59

Our models also share the same variables with different, but still close, coefficients magnitudes. The fact that there are positive and negative coefficients tells us that those parcels must be balanced to meet the target EEA or MER of 0. The coefficients with higher magnitude are the ratio between code reviews and coding, and CMMI level and both contribute to increase the EEA

[4] Tables can be found here: https://www3.nd.edu/~wevans1/econ30331/Durbin_Watson_tables.pdf - last accessed on 29-12-2015

value. The effect of making the code review time lower than the code time is to decrease EEA. To diminish EEA in case of overestimation the variables to adjust are the percentage of the overall time spent on detailed design reviews and percentage of time spent on code inspections reviews relative do the total time spent in development phases. We highlight the fact that to decrease EEA the phases to consider are defect removal phases. The ratio of Code Reviews and Code increases the EEA, which means that the higher the coding time is, in comparison with the time to do code reviews, the lower will be EEA. In our case study CI, when developers knew their code was going through an inspection, the number of defects found could be lower, due o the fact they were more cautious in the implementation, which can also imply spending more time coding than if there was no peer review.

Looking back at the variables in the TSP regression model that contribute to an explanation of the Actual Hours (equation 5.3) we find that some of the variables are collinear with variables that are part of the EEA model, namely the percentage of development time spent in the Code Inspection phase and the ration of time spent in Code Reviews over Code. Interestingly, we also find that one of the variables that increase Actual Hours is Detailed Design, while one of the variables that decrease EEA is the percentage of project time spent in the respective review phase, Detailed Design Review.

We had very few projects of organisations appraised at CMMI, none of them at level 4. We noticed that the organisations with CMMI level 5 had worse EEA and MER. That fact cannot be explained with the data we had, but perhaps those organisations were just starting to use TSP and did not yet have enough TSP historical data.

5.3.6 Cross Validation of the Standard Error

Considering the, already small, number of projects and the fact that not all of them have values for all variables, we decided to use the entire data set to train the model and estimate its error using cross validation of the models instead of using a training and test set, i.e. using part of the sample to generate the regression model and the test set to verify its prediction error.

We created *k*-fold where *k* was 4, resulting in 4 samples, one of them with 22 projects and the remaining 3 with 20. We determined the estimated EEA and MER based on the respective models and used the Standard Deviation (equation 3.8) of the residual error to determine the mean error of the estimates of each one of the 4 folds. The results are summarised in Table 5.14. We did not exclude any projects that did not have planned the phases considered in the models coefficients, which naturally contributes to higher error in the estimate, but also reflects the reality of projects.

The estimates mean Standard Deviation is close to the estimated standard error of the estimate in the models (see Table 5.12), being around 0,03 higher. Using other samples would have a variation of approximately 0,304 and 0,265 from the slope of EEA and MER respectively. As a reference, and because MMRE and Pred(25) are often used to compare the accuracy of estimation models, we also determined their value. The reference values are MMRE <= 0,25 and Pred(25) >= 0,75. The MER model MMRE is within the recommended values to be considered a good prediction model, but Pred(25) is slightly below the value it should have, 0,72 instead of 0,75 or

Table 5.14: 4-fold cross validation of the standard error of the estimates of the models EEA and MER.

Test Set	**SD**		**Pred (25)**		**MMRE**		**Projects**
	EEA	**MER**	**EEA**	**MER**	**EEA**	**MER**	
Sample 1	0,2903	0,2460	0,64	0,68	0,23	0,20	22
Sample 2	0,4336	0,3965	0,60	0,60	0,33	0,29	20
Sample 3	0,3162	0,2703	0,60	0,75	0,25	0,22	20
Sample 4	0,3144	0,2535	0,65	0,85	0,23	0,19	20
Mean	0,3386	0,2916	0,62	0,72	0,26	0,23	***Total***: 82

above. Regarding EEA both accuracy statistics are different from the recommended values, even if MMRE is just slightly above the limit value, 0,26 instead of 0,25.

Even though the MER model is more accurate, having lower error and explaining a higher percentage of the data variability, we opted to display both of them, because if the practitioner wants to distinguish over and underestimations the information is then available in the EEA model.

5.3.7 Differentiating Factors on EEA Quality and EEA Ranges

The EEA model that we built only considered tasks estimated based on size and measured in LOC, excluding all tasks without an estimate of size, and the ones where the size measure considered was other than LOC, as was the example of several requirements tasks that used text pages as measure unit of size. This was a concious decision we made to allow comparable tasks. Besides, the model is based on TSP recommendations, as it includes Process Variables defined at component level considering the recommended values of percentage of time spent in a phase and ratios between defect removal and defect insertion phases. However, organisations build their plans including tasks other than the ones that TSP has recommended values for. Such tasks were also contained in the TSP Database and contributed to the total plan of the project. In the analysis presented in this subsection we considered all projects and their tasks, be they based on size or not and independently of the units used to measure size, which increases our sample to 91 projects. Our purpose was to understand which variables of all those we had available influenced the overall, or total EEA quality. That is to say, we analysed which of their values would minimise the total value of EEA. For that purpose we tested some of the hypotheses we raised in our research. The variable to consider was named Total EEA and is based on all tasks effort of actual and planned hours.

Based on our sample and the previous analysis done of the TSP data we defined quality levels of EEA. In Table 5.15 we compare the data of prior SEI reports of analysis done using TSP data with the data of our analysis. Checking our data details we observed that around 38% of the projects in our sample had an absolute value of EEA below 10%. By analysing the data in Figures 5.14 and 5.15, we defined the values between 10% and 25% as average; an indicator that when estimating effort if the predicted EEA falls in that range (when used as a leading indicator), or after completion the projects are falling in those values, they may consider reviewing missing phases or steps of the estimation process to improve. This top value is also similar to the average effort error measured in the previous analysis (McAndrews, 2000; Davis and McHale, 2003), therefore we set it as an acceptable value that a project using similar estimation methods may be stated as having. When falling in values above 25% the quality of the effort estimation process is weak and there should be a review to improve it. Notice that 200% of EEA should be considered as an extreme, and not be considered in the models. By choice we decided not to remove outliers, as when these cases occur, and they in fact do, organisations should be aware of them.

Table 5.15: Data of previously published TSP reports (McAndrews, 2000; Davis and McHale, 2003) and the analysis we did using the dataset extracted in 2013 (TSP 2013). The medium, minimum and maximum of the effort deviation.

	TSP 2000	**TSP 2003**	**TSP 2013**
Organisations	4	13	17
Projects	15	~20	91
Mean	-4%	26%	18%
Minimum	-25%	5%	-21%
Maximum	25%	65%	200%

The thresholds we identified need to be adjusted by the organisation to consider their goals and the variables that are more important to achieve them considering their business needs and according with the model calibration to their own data. As stated by Nichols (2012, personal communication), the values recommended in TSP are not normally used by organisations starting a first TSP project. Although we would not consider as average value an absolute EEA under 30%. The overrun of software development projects is around 30% (Halkjelsvik and Jørgensen, 2012), so we set our threshold to be under that value, based on the behaviour of the projects in our sample. We recommend that organisations never set 30% as a regular value because having models and additional support systems should help them improve their effort estimates.

We implemented the EEA Model only considering tasks whose parts size measure was LOC. We defined a variable to state if size was used or not. As all of the tasks had size estimates, the rationale to say it was used was that the planned hours were the product of estimated hours and number of engineers, where the estimated hours were the product of estimated size and productivity. The variable had no statistical significance to integrate the model. We now consider a new variable to distinguish tasks that did not have size estimated at all and tasks of parts measured in any size unit. Analysing the projects usage of size in estimation, in none of the projects did the team estimate all tasks without using size, but had mixed tasks: 69% of them estimated tasks both considering size and not considering size. These projects that did both had on average an EEA of 16,28% while the ones always estimating considering size, on average, had an EEA of 23,28%. Therefore, in this sample there is on average an improvement of 6,9% in the estimates when using both methods. In Figure 5.16 we can see the phases that have more tasks estimated without having an estimate of size. Of the total parts of all projects, the size of 80% of them was estimated. We note without surprise that the parts whose size was estimated more than 90% of the time are

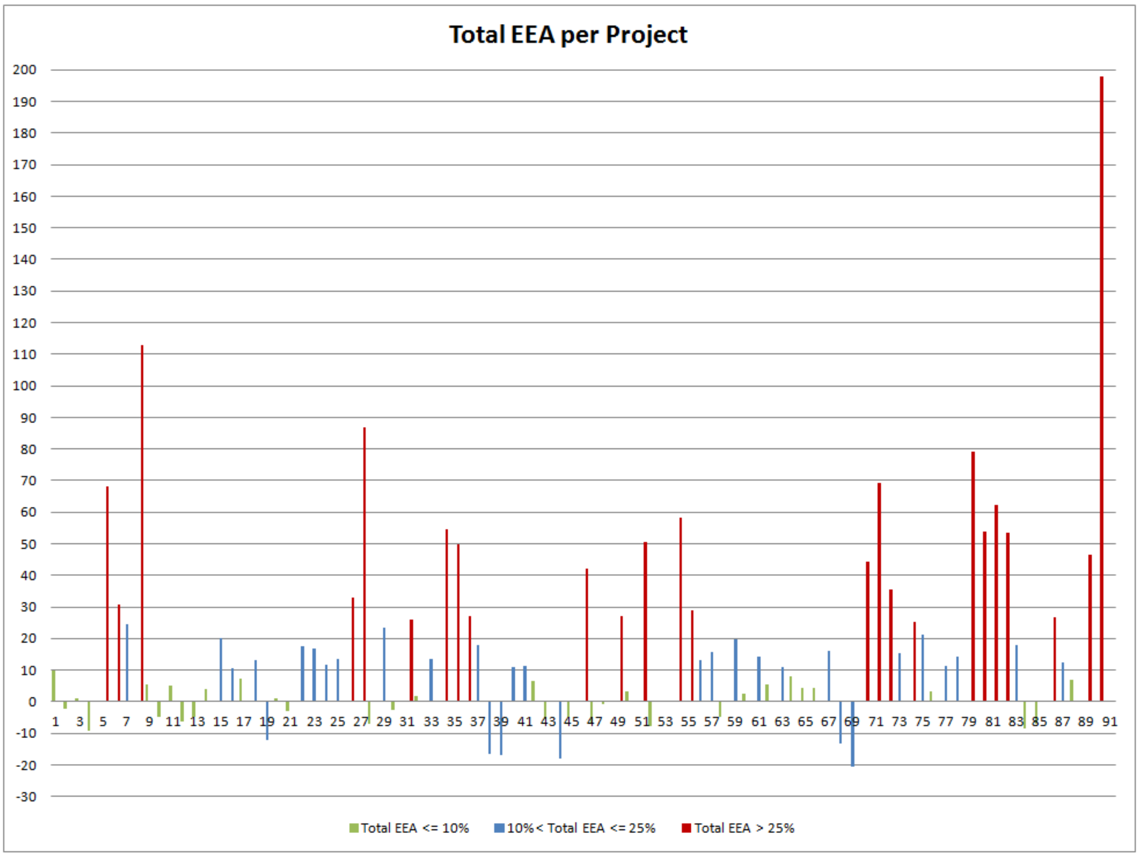


Figure 5.14: Total EEA in percentage, per project.

Note: the project value is coloured according with the percentage of EEA obtained, lower values indicate a better effort estimation process.

code and the related defects removal phases, detailed design and its review phase. Integration and System Test Plans size is also estimated approximately 90% of the time.

Comparing the accuracy of the estimates of phases that were based on size with the ones that were done without size estimates (Figure 5.17), in general the tasks where size was estimated have better accuracy, but there are exceptions. Surprisingly code is amongst the phases where the estimation accuracy presents better results when size is not used and so are requirements and requirements inspections. As we will later see, the size estimates do not present statistically significant improvements on effort estimation accuracy.

Our hypothesis to check if there is a statistical reduction of the Total EEA (V3) mean when planning based on given variables, being a CMMI organisation (V4) or being an HML one (V5), were tested using a one-tailed t-student test (see Table 5.16). The test is valid for normal and non-normal variables when the number of subjects is higher than 30 (Mordkoff, 2010), and being a parametric test it gives us more confidence in the results. The Levene test is once again used to verify if the samples came from the same population, which in this case were all shown to be homogeneous.

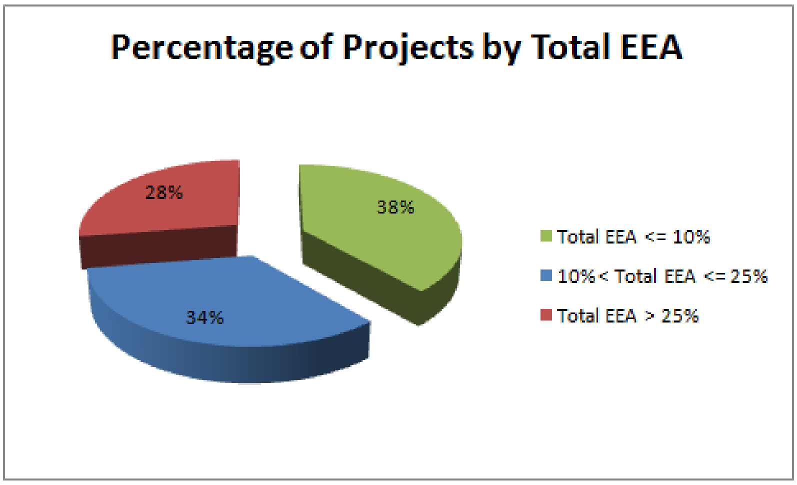


Figure 5.15: Percentage of projects on each Total EEA level.

Note: projects with a good estimation have an absolute Total EEA under 10%; average estimations fall between 10% and 25%; above 25% the value is worse.

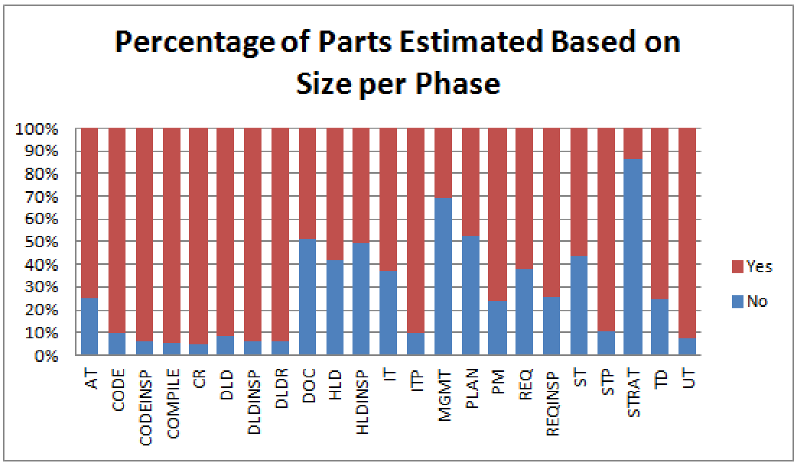


Figure 5.16: Percentage of phases parts that were estimated based on size.

Note: for 80% of the parts, in the sample of projects, size was estimated.

We tested if using size to determine the planned hours improves EEA (V1) and Total EEA (V3) or not. The results show that either using it directly to calculate the plan, based on number of engineers (V2) only having tasks estimated based on size (V6=0) or having better estimates when the size was used (V7=1) than when was not used (V7=0), does not show statistically significant differences on Total EEA. The results of organisations that were rated at HML (V5=1) instead of at low maturity (V5=0) are not shown as significantly better. In fact, even without significance the mean Total EEA of HML organisations was lower. The number of projects with HML was low, therefore we could not expect to get conclusive results when comparing CMMI organisations. Regarding the comparison of Total EEA of organisations that were rated at a CMMI level

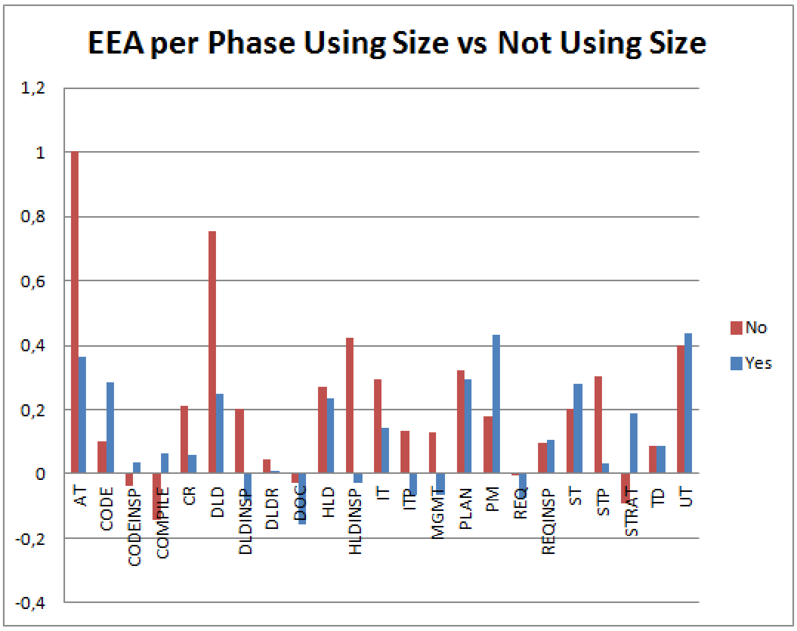


Figure 5.17: Comparison of the tasks per phase where size was estimated and where it was not. Not all phases present better estimates (lower EEA) when size is used.

Table 5.16: Tests between groups to check if there is a statistically meaningful reduction of Total EEA mean when the variable is present (Vn=3).

Variables	Levene	T-Student	Result	Mean

V1: EEA
V2: Size used in planned hours considering planned hours are based on size and number of engineers
	F = 1,729
	p = 0,192
		t= -0,579
		p = 0,282
			Not significantly lower	V2=1 Mean = 0,198571
				V2=0 Mean = 0,260460

V3: Total EEA
V2
	F = 0,216
	p = 0,643
		t = -0,371
		p = 0,356
			Not significantly lower	V2=1 Mean = 0,145463
				V2=0 Mean = 0,173406

V3
V4: CMMI organisation
	F = 2,455
	p = 0,121
		t = 2,385
		p = 0,0095
			Significantly higher	V4=1 Mean = 0,312721
				V4=0 Mean = 0,138971

V3
V5: HML organisation
	F = 0,069
	p = 0,797
		t = 0,561
		p = 0,2925
			Higher but not significantly	V5=1 Mean = 0,395308
				V5=0 Mean = 0,290197

V3
V6: Estimates without size and with size as base
	F = 2,551
	p = 0,114
		t = -0,971
		p = 0,167
			Lower but not significantly	V6=1 Mean = 0,162806
				V6=0 Mean = 0,231791

V3
V7: Estimates with size that have better results
	F = 2,424
	p = 0,125
		t = -1,373
		p = 0,0875
			Not significantly lower	V7=1 Mean = 0,118538
				V7=0 Mean = 0,225802

V3
V8: Plan DOC
	F = 0,818
	p = 0,368
		t= -2,034
		p = 0,0225
			Significantly lower	V8=1 Mean = 0,120644
				V8=0 Mean = 0,251742

V3
V9: Plan ITP
	F = 1,658
	p = 0,201
		t= -1,856
		p = 0,0335
			Significantly lower	V9=1 Mean = 0,120644
				V9=0 Mean = 0,251742

Legend: DOC - Documentation phase, ITP - Integration Test Plan phase.

(V4=1) with the ones that were not (V4=0), it was shown that this sample presented a statistically significant difference of means (at a confidence interval of 95%). Surprisingly, and as was shown in the EEA model, the organisations appraised at a CMMI level have higher Total EEA, hence having higher deviations of their estimates. We noticed that one of the organisations without a CMMI certification had a considerable number projects in the sample, therefore we studied if there were differences between organisations and differences between this organisation and all the others. The results did not show an improvement of Total EEA when an organisation had more projects. We tested it because we considered that this particular organisation could have a more stable TSP process because of the higher number of projects. Finally, we tested whether there was a difference between having planned given phases other than the development ones. The results had shown no difference of Total EEA when considering them except for the projects that planned for documentation (V8=1) and an Integration Test Plan (V9=1). These had lower Total EEA than the ones that did not consider those phases in their plan.

We also analysed the effects of our continuous variables that we used to build the EEA model to check if there are differences between their means across the levels of Total EEA, in the ranges defined previously. We used the non-parametric test Kruskal-Wallis, as none of our dependent variables follows a normal distribution. We compile the variables that showed differences of their means in different levels of Total EEA, in Table 5.17. In the case of the CMMI levels, we used the median instead of mean. The hypothesis of the test are:

H_0- The distribution is the same across categories

H_1- The distribution is different across categories

A different distribution indicates that for those variables there is a range that better represents them. We also wanted to see which EEA of the phases could be distinguished in the ranges.

In the case of CMMI levels we used the medians. For those cases, even though the distribution is different across categories there are more than 20% of cases with CMMI level lower than 5 and there is no case available of an organisation appraised at ML 4. This variable and the percentage of time spent on DLDR are part of the EEA model we developed. DLDR percentage was also shown to have a different distribution between Total EEA ranges. We can observe in the variable boxplot distribution by Total EEA Range (Figure 5.18) that as DLDR percentage lowers the range of EEA is worse. DLDR percentage in the implementation phases also presents a similar (Figure 5.19) tendency but are harder to distinguish.

Regarding the projects that followed the guideline for the ratio of Requirements Inspections and Requirements, when the guideline is followed the Total EEA is below or equal to 25%, but there are cases when it is both either followed or not followed that have a worse Total EEA, greater than 25%. There were few projects in the dataset with data for these two phases.

We include the box plots of the EEAs of each of the phases that have distinguishable distributions per each Total EEA Range in appendix C. The distribution between ranges is better defined when considering the confidence interval of 99%, the boxplots per range do not overlap as much.

Table 5.17: Variables whose means distribution is different across Total EEA levels. The default confidence interval (CI) is 95%, we indicate the ones where it is 99%.

Variable	**Kruskal-Wallis**
CMMI	p = 0,0038 CI: 99%
DLDRPerc	p = 0,0040 CI: 99%
DLDRPercDev	p = 0,0480
DLDINSPAllocPerc	p = 0,0350
REQINSPREQFollowed	p = 0,0280
EEA of the phases that are better distinguished between ranges	
CODEEEA	p = 0,0000 CI: 99%
CODEINSPEEA	p = 0,0450
COMPILEEEA	p = 0,0060 CI: 99%
CREEA	p = 0,0020 CI: 99%
DLDEEA	p = 0,0040 CI: 99%
DLDINSPEEA	p = 0,0340
DLDREEA	p = 0,0180
ITEEA	p = 0,0070 CI: 99%
TDEEA	p = 0,0040 CI: 99%
UTEEA	p = 0,0000 CI: 99%

Legend: DLDINSPAllocPerc - percentage of components that followed DLD Inspections Allocation guideline, REQINSPREQFollowed - percentage of components that followed recommended ratio between time spent implementing Requirements and inspecting them, DLSINSP - DLD Inspection, IT - Integration Tests, TD - Tests Design, UT - Unit Tests.

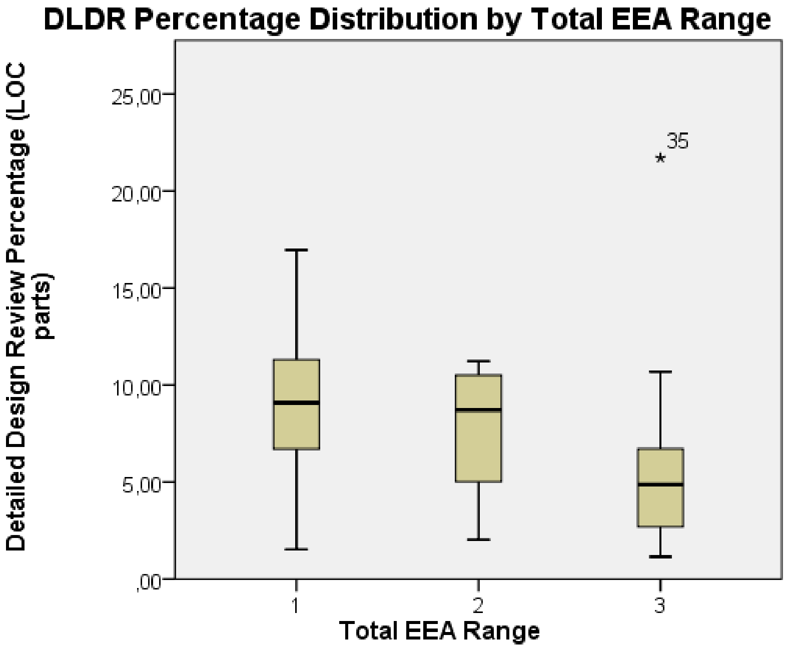


Figure 5.18: Detailed Design Review Percentage (DLDRPerc) of tasks where the parts were measured in LOC distribution by ranges of Total EEA.

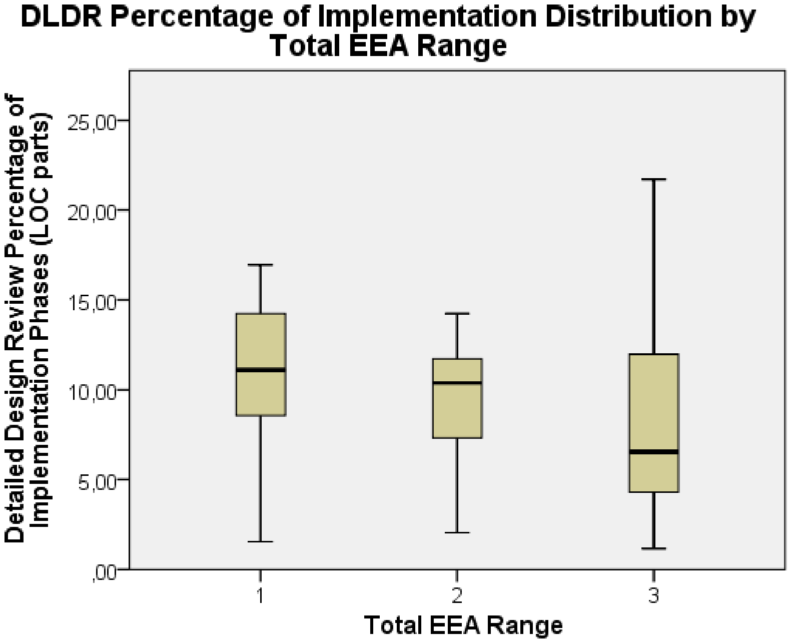


Figure 5.19: Detailed Design Review Percentage of the Implementation Phases (DLDRPercDev) of tasks where the parts were measured in LOC distribution by ranges of Total EEA.

Legend: in the x axys **1 -** Total EEA <= 10%; **2 -** 10 < Total EEA <= 25%; **3 -** Total EEA > 25%.

We posed a set of hypothesis to analyse the effects of the variables we considered that would be relevant to EQualPI when evaluating the performance of the effort estimation process. We revisit them in Table 5.18 where we include our results and some of those obtained by other authors as well.

Table 5.18: Hypotheses related to the value of the Performance Indicator Effort Estimation Accuracy: results that we obtained in our research and other authors' results. The ones not signalled were testes by Morgenshtern (2007).

Hypotheses	**Our Results**	**Literature**	**Results**
Related with the process			
HA- There is a difference between organisations of different CMMI levels, organisations with higher CMMI levels have better EEA	The variable CMMI is one of the coefficients of the EEA model. We saw that increasing the CMMI level the EEA became higher, as opposed to improving it.		Worse estimates at higher ML.

Continued on next page

Table 5.18 – *Continued from previous page*

Hypotheses	**Our Results**	**Literature**	**Results**
HAA- Organisations using CMMI have better EEA than the ones not using CMMI	The Levene test showed that the samples variance was homogeneous. When comparing total EEA, the t-test showed that the mean was significantly higher at organisations rated at a CMMI level. The analysis to the variable of having or not CMMI revealed a difference in the total EEA means. The organisations with CMMI, consistently with the previous hypothesis, were lower.		Worse estimates at CMMI organisations.
HAB- Organisations rated at HML have better EEA than organisations rated at LML	We tested whether there was a difference in the means of total EEA between organisations at HML and at LML and did not find statistically significant differences.		No statistical difference.
HB- Different organisations using the same estimation methods present different EEA results HBB- Organisations with more experience (number of projects) using the same estimation methods have better estimation results	We did not find differences between the organisations. Even in organisations with more projects, that could have improved their process in time, have improved their performance or know them better.		No statistical difference.

Continued on next page

Table 5.18 – *Continued from previous page*

Hypotheses	**Our Results**	**Literature**	**Results**
Estimation variables			
HF- Estimates based on size improve the estimation accuracy	We could not find the influence of estimating size on the effort estimates. Nonetheless, we found that the estimates of size in these projects had a poor R square, so for that reason their size estimates were still improving.		No statistical difference.
HH- Projects that estimate other phases based on size that are not code have better effort estimates (HLD, Requirements, DLD...)	We found several variables other than code that contributed to the Total EEA and were part of the EEA Model.		Several other phases influence EEA.
Comparison			
HJ- Projects that use historical data produce better estimates	If for this hypothesis we considered the projects not estimating size, as in that case similar projects had been done in the past, then we did not find a relation. We cannot draw any conclusions based on our results.		Inconclusive.
HK- Using benchmark data improves the estimates when no other source is available	If we consider that when a guideline is followed the TSP benchmark is used we saw a statistically significant difference between following the REQINSP (Requirements Inspection) over REQ (Requirements) recommended ratio and not following it. The data is insufficient to conclude about this hypothesis.		Inconclusive.

Continued on next page

Table 5.18 – *Continued from previous page*

Hypotheses	Our Results	Literature	Results
People experience			
HLA- Experts' experience in number of projects improves the accuracy of the estimates	We only have context data of 5 projects.	H13A – Estimator with higher number of projects in the specific application have better estimates	Lower duration error. Estimates get better when managing many small projects instead of fewer larger ones over the same number of years.
HLB- Experts' experience in number of years improves the accuracy of the estimates	We only have context data of 5 projects.	H13B – Estimator with more years of experience	Lower duration error but not significantly.
HM- Planning done with the elements participating in the project improves the accuracy of the effort estimates	Assumed to be always true in TSP as the team members participate in the planning meetings.	H9 – External estimators provide lower estimates than the developers, Lederer et al. (1990)	Based on our assumption and other authors results we would conclude it leads to small estimation errors.
		H13C – Trained in IT project management compared with untrained estimators	Smaller estimation errors.
		H13D – Training in estimation techniques	Better duration estimates.
Estimation process and project phases considered			
HN– Granularity of tasks improves tasks stability and consequently effort estimation accuracy	We did not test this hypothesis.	H5 – Complex tasks insufficiently broken down lead to underestimation errors, Hill et al. (2000)	Low granularity complex tasks increase underestimation errors.

Continued on next page

Table 5.18 – *Continued from previous page*

Hypotheses	**Our Results**	**Literature**	**Results**
		H11 – Projects with higher use of estimation development processes exhibit smaller duration and effort estimation errors	Partially confirmed. Number of WBS levels is uncorrelated with estimation errors. Shorter activity durations and smaller task efforts give smaller effort estimation errors. Better estimation goals definition give smaller estimation errors (greater impact on duration than on effort). Better estimation techniques and commitment processes give smaller duration errors but not significantly.
		H12 – Projects with higher use of estimation management processes exhibit smaller duration and effort estimation errors	Partially confirmed.
		H12B – Frequent reporting H12C – team performance assessment H12D – risk management	H12B and H12C and H12D, reduce duration errors. H12B, leads to higher commitment and better estimations.
HO– Re-estimation at the beginning of phase improves the effort estimation accuracy	We did not test this hypothesis.	H12E – Frequent work plan updates improve estimates	Better duration estimation.
HQ- Designing the architecture improves the estimation accuracy	Considering that the architecture would be part of High Level Design we could not show its influence on the EEA.		No statistical difference.

Continued on next page

Table 5.18 – *Continued from previous page*

Hypotheses	**Our Results**	**Literature**	**Results**
HR- Doing detailed design improves the estimation accuracy	The percentage of the time spent on DLDR was one of the EEA model coefficients and shown to have an influence on the total EEA. Therefore, reviewing the design improves the estimation accuracy.		Better estimates.
HT- Quality phases improve the estimation accuracy	DLDR, Code Inspections and the ratio between CR and Code are coefficients of the EEA Model. Following the recommended allocation to DLD Inspections was shown to have better total EEA. Having the phase *per se* did not show improvements of estimates, but the accuracy the estimates of those phases seemed to make a difference in the total EEA (Table 5.17)		Better estimates.
Project characteristics and execution			
HWA- Project duration improves the effort estimation error	We did not test schedule hypotheses.	H2 – Large variance in time and budget lead to estimates bellow actual values, Hihn et al. (1991) H10C – Estimated Duration (tested with H10D)	H2- Large time and budget lead to lower estimates than the actual. H10C- for larger and complex projects the duration estimation errors are smaller than the overall duration.
HWB- Project complexity increases the effort estimation error	We only have context data of 5 projects.	H10 – Projects with higher level of uncertainty exhibit larger duration and effort estimation errors	Partially confirmed. Project size increases duration and effort errors.
		H10B – Complexity	Increase duration and effort errors.
		H10D – Implementation Complexity (tested with H10C)	For larger and complex projects the duration estimation errors are smaller than the overall duration.

Continued on next page

Table 5.18 – *Continued from previous page*

Hypotheses	**Our Results**	**Literature**	**Results**
		H4 – Uncertainty reduces estimation accuracy and hence the project performance, Nidumola (1985)	Not quite comparable because a project may be complex not because of uncertainty.
HWC- Solution novelty increases the effort estimation error	We did not test this hypothesis.	H10A – Innovativeness of need	Increase duration and effort errors.
HWE- Team size increases the effort estimation error	Team size did not influence total EEA and was not a coefficient of the EEA model.	H3 – Project size affects time estimations Brooks (1982)	In small projects communication is tighter and problems are easily dealt with, Brooks (1982).
HWF- Team experience improves the effort estimation accuracy	We only have context data of 5 projects.		Lower team experience increases duration estimation error.
HWH – Closeness of relations with the client improves the effort estimation accuracy	We could not test this hypothesis.	H12A – Higher customer control reduces effort estimation error	Reduce estimation error.
		H12F – Customer control (via steering committee) reduces relative Effort estimation errors, more correlated with duration due to the focus on duration management rather than effort management	Higher effect on duration rather than effort error.
		H7 – Good relation between developers and the customer have a positive effect on estimation accuracy, Van de Ven and Ferry (1980)	Better estimation accuracy.

Continued on next page

Table 5.18 – *Continued from previous page*

Hypotheses	Our Results	Literature	Results
		H6 – Managerial control improves estimation accuracy, Van de Ven and Ferry (1980)	Better estimation accuracy.
		H8 – Sense of responsibility and commitment contribute to estimation accuracy	Better estimation accuracy.

With our dataset, we found there is a negative influence of the CMMI level, being related with higher EEA values, consistently with the EEA model. We found a positive effect of having planned time for defect removal and detailed design phases in the EEA result. The team size showed no statistically significant influence on the EEA indicator. Finally, we were expecting that when using size the EEA would be better, but this was not the case. Even though size was not shown to improve effort estimation accuracy, with the data we had, we encourage its usage in order to have a better notion of the effort used, and start building better models for size prediction. In fact, in the case of the data we used, the linear regression model of the added and modified code size estimates has low R square (24% and 59%, in the logarithmic scale), which means there is still room for improvement at least in the considered organisations, as can be seen in the graph in Figure 5.20. This fact indicates that these projects are not yet using a Probe B method, where plan and actual are required to correlate with R greater than or equal to 70% (Humphrey, 2005).

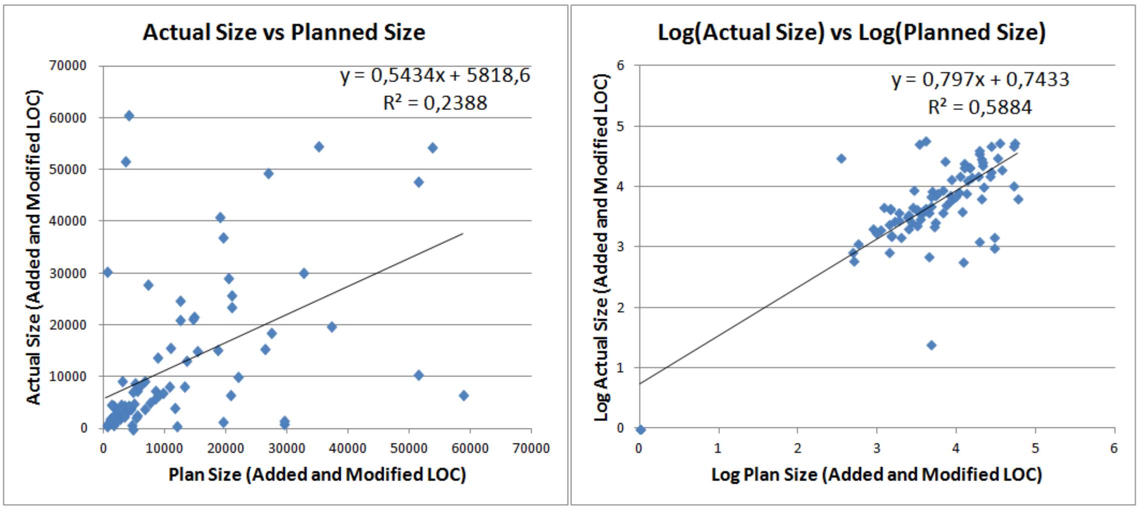


Figure 5.20: Regression model of actual size and planned size. At the right in logarithmic scale.

5.3.8 Limits to Generalisation and Dataset Improvements

The models we built would be more complete if we had a sample with more projects, where the different TSP planning procedures had been followed and all phases were included. When checking the variables correlations we noticed that Requirements, Requirements Inspections, High Level Design and High Level Design Inspections as well as the corresponding Process Variables would be relevant to this research. Furthermore, even though we are aware that it is harder for people to accurately estimate injected and removed defects (especially injected), the model could have considered such estimates, if we had had the data. However, having just a single data point does not allow us to reach any conclusions. These variables should be considered because when executing the project the actual defects have influence on the quality of the product and require time to be fixed, affecting the actual hours. If the data is available, new Process Variables should be designed based on phase yields, percentage defect free, defect density, defect injection and removal rates, as defined by Humphrey (2006).

We introduced researcher bias when we excluded projects with plan and actual hours equal to zero, respectively. However, some tasks may not have been planned and still had to be added to the records and executed; similarly, some tasks may have been planned but may not have been completed. This decision was consciously taken as the status of the workbooks may not have been the final (end of project), therefore we could not be certain of the reasons for the plan and actual values of those tasks to be 0 and whether they should be kept or not.

We found several missing values and different case/naming for the same values that could have been avoided by providing dropdown lists, auto-complete and some automation. For project execution purposes it may not have been a problem, as engineers already knew the meaning of the information, but the analysis of the workbooks for Quality Reviews and now for research purposes could benefit from ensuring mandatory information and auto-complete whenever possible. It is important to use uniform phase names and size measures to ensure they are not named slightly differently, affecting posterior data analysis. It is also important to ensure people understand the definition of the phase names and size measures to avoid the creation of new ones due to lack of understanding.

We also reviewed the workbooks to identify those that were a different version of the same project. Currently there is no workbook versioning in the database; each version gets a new workbook ID. It is important to ensure the database has a versioning mechanism of workbooks, so when an existing one is updated, instead of receiving a new ID, it gets a new version, time stamped, in order to allow sequencing. In this manner researchers can choose to analyse the same project over time or only analyse the latest version.

5.3.9 EEA PI Discussion

We defined a regression model useful to analyse the quality of the effort estimation process. It was first conceived for the CMMI PP SP 1.4, based on data collected from TSP projects, but ultimately it can be applied to any effort estimation process, this just being a matter of changing the rules used to estimate and the recommended values used for estimation.

The use of TSP data relies on the use of size which fits the guidelines of PP SP1.4. In particular, several TSP rules are designed for LOC based projects, namely rates per hour and review rates per hour. This should not be a reason for organisations that do not use such a size measure to consider this research work inapplicable to them; on the contrary, there are several adaptations that can be done and also several recommendations that are still applicable:

- Size can be measured in any unit that fulfils their needs, as long as it is consistently measured in all projects. Since the goal of any process is to use the organisation's historical data, after a cycle of development there will be a dataset to begin understanding what their own development rates are and, with time, recommended values;

- Percentage of time spent per phase is independent of the size measure, any means used to determine the effort needed for a phase will allow computing of the time per phase of the others;

- Review rates can also be adjusted in time by defining what optimal time should be spent reviewing a given work item to gather a relevant number of defects (which can give confidence that yield will be reduced).

Another reason for choosing TSP data is the fact that it is consistently collected by different organisations. Many TSP metrics are common to other methods and are defined to plan and follow the development of software. The number of factors that we can monitor with TSP metrics makes us believe that we can build a more complete model.

EQualPI provides a Data Dictionary to facilitate the collection of the base data, either by us or the SEI, in order to define the set of variables that could be used to characterise EEA and build an EEA model that could give organisations information about the quality of implementation of their estimation process upfront, that is, before completing the estimation process and starting to develop the product. The variables used are based on inputs and outputs of the estimation process and development process, when estimating product size, task duration and injected and removed defects. Of all variables defined in the Data Dictionary we were only able to gather information on part of them and, in other cases, only had access to a limited number of data points, which did not allow us to test all our hypotheses. Another difficulty lies in not having certain properties identified in the database. We wanted to test if historical data would result in higher process quality, but as we did not have any variable that could explicitly tell us whether the task was estimated based on historical or benchmark data or not, we could not reach a definite conclusion on this hypothesis.

The base measures used are the ones defined in TSP, such as estimated size, size unit, time and phase. Those were the variables that allowed us to define the derived variables and, as indicated in EQualPI, do aggregation at different levels; the Domain Model allowed us to better understand the relationship between variables and design the aggregation procedures. Nevertheless, and as indicated in EQualPI, the aggregation is not reduced to a sum of parts, it has additional complexity, based on the purpose of using a variable in particular. In our case we had to define how to aggregate tasks information, to the component level, and from the percentage of compliance at component level define the percentage of compliance at project level. We had to realise which PIs allowed us to understand the estimation process, which in this case was the TSP estimation process; which indicators contribute to plan the project; and what was the output of a model based on using, or not using, those indicators. We could not include all variables, leaving out the experience in the estimation and development process (prior training in TSP/PSP), influence of interruptions, and estimates of injected and removed defects, for example.

In the definition of the model we used three process variables:

- Percentage of time spent in phase in the overall phases;
- Percentage of time in implementation phase in the overall implementation phases, as defined by Humphrey (2006);
- Process PIs, in this case based on the TSP guidelines.

We could have analyse other compliance variables to understand their influence on EEA and whether they should be part of our model or not. However, we did not have them in the database, nor access to the organisations who contributed to the TSP database, in order to know the process followed. If we did have more information we would analyse the size method used: ProBE A, B, C or D; if the estimates of size were based on proxy's with historical data, historical data of similar projects, or historical data of similar components; if the development rates were based on historical data and similarly for other tasks rate, for example. We also would include variables related with questions such as *What was the process used?*, *What was the sequence of steps done to estimate?* and *Were the estimates done individually and then discussed and aligned to build the final plan?*

EEA determined by the deviation of the plan from the actual hours itself would just be a lagging indicator of the performance of the effort estimation process, measuring its effectiveness, considering that the objective of estimating effort is to produce a plan that is close to the execution. What EQualPI aims to achieve is to give organisations PIs to act on as well. We built the EEA model to provide a means to anticipate the process effectiveness, hence the model is a leading indicator of the performance of the effort estimation process (CMMI PP SP1.4 "Estimate Effort and Cost") composed of lagging indicators produced when executing the process, that organisations can act on and adjust before starting a development cycle. The indicators that we defined based on the TSP guideline values, that would be compliance PIs, were not significant and consequently are not part of the EEA model. However, the variables that are based on the percentage of time in phase when compared with the overall plan or just the implementation phases, and the ratio between defect insertion and defect removal phases are significant and thus part of the model. Either the organisations use their historical data, or the compliance guidelines were ignored, as only one project demonstrably followed all guidelines, and following guidelines were not variables included in the model. We consider that if in fact organisations are actually using their historical data, this justifies the accuracy of their estimates, when looking at the TSP model of Actual as a function of the Plan hours (equation 5.2). Therefore, the usage of benchmark values to begin with, and as historical data becomes available use that data to calibrate the models, improves their accuracy. The indicators influencing Total EEA, and the tested hypotheses, also provide a useful set of PIs that organisations can have in place and act on to manage, make adjustments when instantiating the process in a project, and to improve the estimation process. With the variables used to define the EEA model we answered research question **RQ5** - *Is it possible to define metrics to evaluate the quality of implementation of CMMI practices focused on their effectiveness, efficiency and compliance?*

The EEA model we developed has an accuracy of ~30%, which could be considered a small Adjusted R Squared for a prediction model. However, this model is predicting a deviation of the actual effort and the estimates, the error of the estimate itself. Therefore, we consider that using it along with the other PIs identified in this chapter will truly help organisations evaluate the quality of implementation of their practices. The 30% of variability, explained by our model, represents the percentage of EEA that organisations can act on, in order to, before completing the plan, improve the quality of their estimation process. Although, the remainder 70% include in fact the variables we could not study, and that might be part of the model, the measurement error, along with the uncontrollable factors. This helps us to partially answer research question **RQ6** - *Can we determine the effects, expressed in a percentage, of uncontrollable factors in an evaluation metric?*

With the variables we identified and following a similar procedure to build the model, organisations will be able to build their own EEA model using their historical data. Moreover, the thresholds – values we defined for the Total EEA ranges – should be adjusted according with the organisations business goals. We are aware that the use of more complex prediction methods, other than multiple linear regression, to build the model would lead to higher adjusted R square. Nonetheless, we wanted to ensure that our results would be interpretable and easily understood, when putting them in practice, and deciding how much variation in a coefficient needs to be applied in order to improve, and hence reduce, the EEA value. Furthermore, other data sets can reveal different variables of influence depending on the phases that the organisations consider when planning, and the emphasis they put on them. If the model we designed was focused on schedule, we sense that the deviations would be even smaller than the deviations found on effort estimation, as teams can put more effort when committed to meet the plan dates, and project managers at times increase effort to meet the schedule even though it will increase cost.

Of the EQualPI modules we developed, we validated six of them along with the metamodel and the Framework principles, by performing literature reviews, case studies, field experiments, modelling, data analysis and statistical tests. We present the validated components in Figure 5.21 in green (brighter shadow).

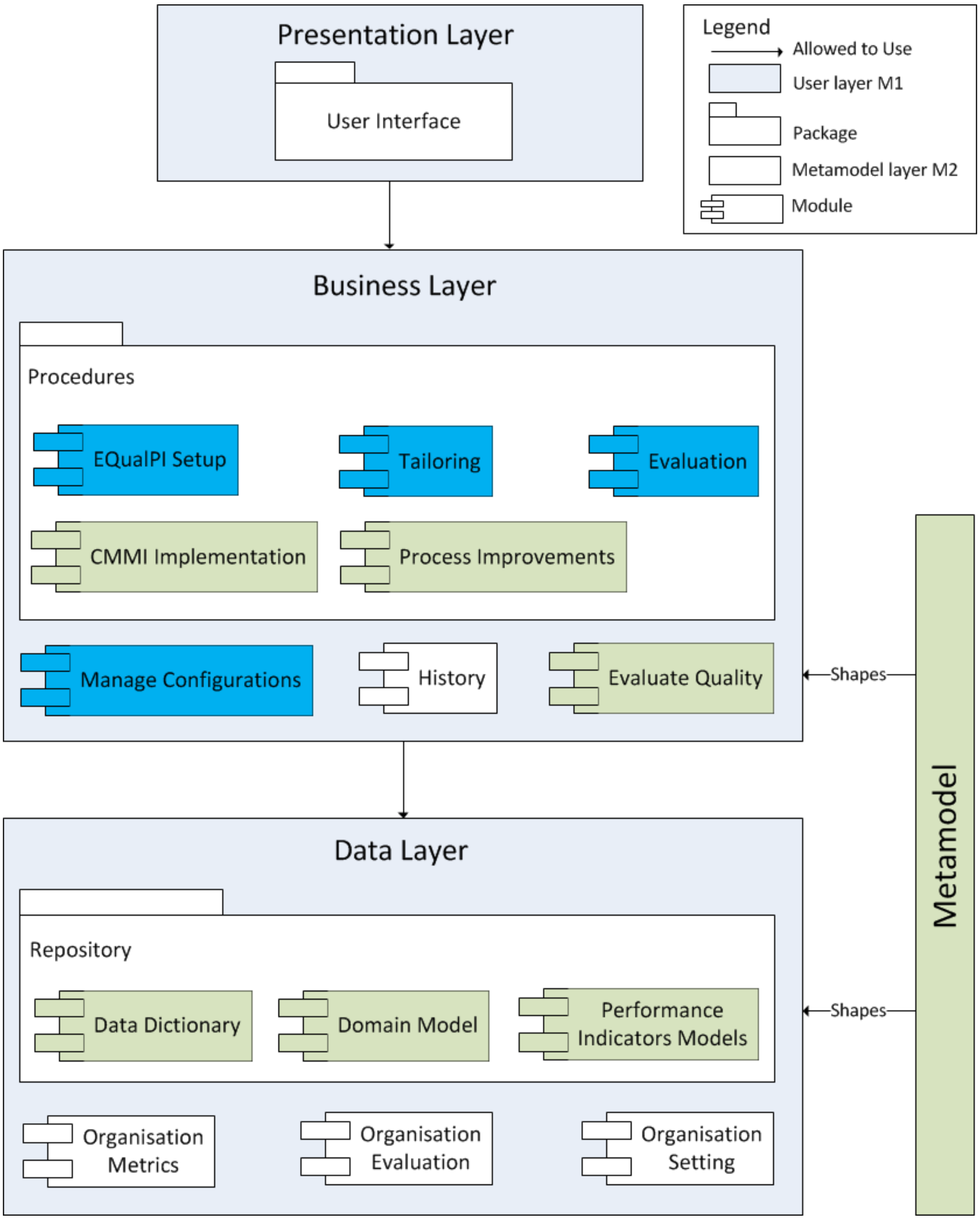


Figure 5.21: Modules of EQualPI that we validated signalled in brighter blue/shadowed, including the metamodel.

Chapter 6

Conclusions

Concerning CMMI problems, the DoD expressed the necessity to "Develop meaningful measures of process capability based not on a maturity level, e.g. Level 3, but on process performance" (Schaeffer, 2004). CMMI V1.3 is more focused on the performance of organisations but SCAMPI is becoming more efficient (Phillips, 2010a), as it reduced the amount of necessary evidence – eventually increasing the probability of leaving problems undetected. In this research we developed a framework to evaluate the quality of implementation of the CMMI practices, EQualPI. The evaluation is based on the quality of outcomes, and to demonstrate the Framework we built a performance indicator model to evaluate CMMI PP SP1.4 "Estimate Effort and Cost".

6.1 Research Achievements

The difficulties in implementing CMMI, in particular HML, are common to the problems found on metrics programmes and software process improvements in general. In particular, Software Engineering metrics can be ambiguous (Goulão, 2008; Breuker et al., 2009), preventing an implementation common to all organisations. With the objective of understanding CMMI problems better, we conducted a literature review, three case studies and further analysed the data of a survey conducted by the SEI. We compiled a list of problems and recommendations to help organisations implementing CMMI, and additional recommendations to support them on the choices to be made regarding the implementation of MA when aiming for HML. Interestingly enough, part of the identified problems is rooted in the CMMI lower maturity levels (2 and 3), as they must be stable before moving to high maturity. The evidence we analysed show that the problems were common to the different sources of information. For the ones we found in the case studies, we verified that they also occurred in other organisations. Furthermore, there are also several problems in the Measurement and Analysis process area that become more evident when implementing CMMI maturity level 4, as they affect process performance models and baselines.

With the conducted case studies we added problems to the ones in 3.1.3 Problems in Process Improvements, Metrics Programs and CMMI and 3.2 Survey on MA Performance in HML Organisations: copied processes, multicultural environments, imposed processes, baselines not applicable to all projects, abusive elimination of outliers, effort estimates, tools setup, overhead and tools requirements. We consider that with a more complete analysis of CII and CIII we could have found more problems.

There is a wide variety of methods to implement CMMI practices. As the model is just a guide telling what to do, but not how to do it, room is left for various implementations that may not always lead to the desired performance results. Moreover, SCAMPI's objectives do not include appraising performance. Consequently, problems and difficulties can occur when implementing CMMI, some of which can persist after appraisal. EQualPI is a framework for self-assessing the quality of implementation of CMMI practices and effects of improvements, based on compliance, efficiency and effectiveness (Lopes Margarido, 2012a). The Framework helps to prevent implementation problems and allows better control of organisations performance. We defined EQualPI's architecture up to level 2, defining all its layers, shaped by a Metamodel, their included packages and corresponding modules. The Metamodel establishes the alignment between the EQualPI repository and the CMMI Goal or Practice. While the methods in the repository support the implementation of the Goal/Practice, the Performance Indicators quantitatively evaluate its achievement. The evaluation is done by aggregation depending on the source (project, department or organisation) or target (PI, Method or Goal/Practice).

EQualPI already includes a module to manage configurations to consider continuous improvement of the organisation processes and allow piloting and the progressive deployment of process improvement initiatives, also evaluating their impact. Once the Framework is completely populated with the organisations processes, methods and performance indicators and a change is piloted, the effect of that change on other practices can be measured.

The package procedures includes how to setup the Framework, select the methods and perform an evaluation of practices. A checklist of recommendations is included to avoid problems in the implementation of CMMI. Those resulted of the semi-multiple case study[1] we did on CMMI HML organisations, as we could not apply the same design in the three case studies, and analysis of a survey of organisations implementing MA aiming at achieving CMMI HML. Based on a literature review, on the survey technical reports and applying statistical methods to that survey data, we gathered a set of recommendations to have high maturity measurement and analysis, which can be used on any measurement program. We also provide steps to consider when carrying out process improvements, aligned with the scientific method. Part of those steps were validated in a process improvement in the requirements review process. From that improvement we assembled a defect type classification list specific for requirement defects that is now used by an HML organisation. Practitioners can use the model to improve the Effort Estimation Model by acting on the variables of the model, that is reviewing the planned values estimated, before moving to the development phase of the project.

The repository includes the Data Dictionary, Domain Model and a Performance Indicator Model to evaluate the quality of the practice "Estimate Effort". The Data Dictionary and respective Domain Model are already complete enough to include the effort estimation and the

[1] As we argued in subsection 5.1.3 Problems Analysis and Limits to Generalisation.

development processes. The EEA model allows organisations to anticipate the accuracy of the effort estimates achieved when estimating their projects, and to act on certain factors to improve that accuracy. Kitchenham et al. (1995) stated that the model error is the sum of the model incompleteness and measurement error. In fact, our model explains approximately 30% of the variability of EEA, meaning that the remainder is related to measurement error and other variables not considered in the model, including the ones related to project execution and uncontrollable factors. We did a regression model of the actual hours estimated using TSP and the accuracy is already high, leaving only 7% unexplained. The 30% of EEA that the variables explain shed some light on the percentage of the actual hours that the used methods did not explain.

6.2 Validation Discussion

Regarding the CMMI Implementation procedures; the list of problems and recommendations is common to problems found by other researchers, the SEI surveys data and in the case studies we performed, so it showed its usefulness as the problems occurred and some of the recommendations were also followed. Additionally, we applied some of them when building the EEA model. Regardless we consider that the design of the first case study should have been used in all case studies with the same extent, and this could reveal more common problems and even new ones. That setting would also provide us more information to support our findings.

The Process Improvements procedure is also aligned with the EQualPI principle that process improvements must be evaluated based on the quality of implementation, that is to say, based on the effectiveness of the outcome when compared to the baseline, and designed to achieve a business goal. When comparing the effects of the improvement with the measured baseline and while monitoring the performance indicators of other related practices, the organisation is ensuring the benefits will be effective and there will not be a degradation in the performance of related processes. The list of recommendations when implementing CMMI is applied in some of the Process Improvement Steps:

- **Step 2 - Identify and define improvement** and **Step 3 - Determine selection criteria and select improvement methods accordingly**
  R7. Involve experts and users of the processes;
  R21. Do gradual data collection;
  R26. Quarantine outliers that are not understood (if needed);

- **Step 4 - Set improvement goals and how to validate them**
  R16. Have measurement reflecting goals;
  R17. Use measurement and analysis protocols;

- **Step 5 - Pilot the process improvement**
  R21. Do gradual data collection (may be necessary);
  R26. Recognise special causes of variation (may be necessary);

- **Step 6 - Analyse pilot results** and **Step 7 - Prepare final version**
  R15. Mature processes and metrics with practice;

- **Step 8 - Progressively deploy and control**
  R12. Have complete and adequate training;
  R33. Improve tools with usage;
  R34. Let tools and processes stabilise before considering the collected data;
  R35. Guarantee collection process precision;
  R13. Do projects and people coaching (if needed);
  R14. Top management: set goals, plan, monitor and reward (if needed);
  R36. After PPM and PPB stabilisation only collect necessary data (when applicable);
  R37. Use automated imperceptible data collection systems (when applicable).

In building the EEA model, we applied EQualPI's data aggregation and normalisation principles that allowed us to have comparable performance indicators amongst different projects. We did not have the same data in all projects, because each project has its own duration, but the data can be normalised by considering percentage of time spent in phase, for instance. The principles applied in the EQualPI framework to build performance models and baselines, as defined in CMMI and analysed in the SEI surveys data were applied. While extracting the data to build the EEA model and check which variables influenced the Total EEA, we followed some of EQualPI CMMI Implementation recommendations. This was done at varying points:

- Defining our variables;
- Collecting and cleaning the dataset to remove duplicates and inaccurate data points;
- Distinguishing missing data from zeros;
- Checking the data for unusual patterns which also allowed us to detect workbooks that should not be included in the dataset;
- Involving a statistician for guidance in building and interpreting the regression models;
- Involving SEI experts, including in TSP, in the data extraction and interpretation while building the models.

The EEA quality can be evaluated considering efficiency, by measuring the effort spent planning to obtain the desired effect, and effectiveness, namely when having low EEA values. In other words, quality is evaluated by the quality of the process outcome, as prescribed in EQualPI.

We found a set of controllable factors that can be acted on to improve the EEA, by decreasing its value and approaching it to the ideal value of zero:

- Ratio between time spent in Code Reviews and Coding;
- CMMI level;

- Percentage of the Implementation time spent on Code Inspections;
- Percentage of total project time spent on Detailed Design Review.

Even though CMMI level is one of the coefficients of the model, there should be more data points of the different CMMI levels in order to sustain the conclusion that with the increase of the level EEA also increases. We consider that by applying a similar approach to build the model, calibrated with another data set, other variables that were not so relevant in our model would be revealed; even different indicators related to the different estimation methods in use could emerge. Additionally, and as seen in the conducted case studies, different projects natures and phases durations would reveal different variables of influence, and require specific EEA models, as was the case of CII. That is why it is important that organisations calibrate the EEA model to their data once they have them and adjust the EEA ranges to their own performance and goals.

The Data Dictionary we built had the right information; including a measurement protocol is useful for the organisations to know where to extract their data from, get their variables definition documented and consequently updated, so they are useful for the ones using them. When implementing the model the Data Dictionary was useful to do the metrics extraction, but we still faced the challenge of creating a table that would include all variables per subject and then evolve it to the different levels of aggregation (component and project) in order to have meaningful results. We also had to define the variables related to the process itself (following process guidelines).

The Domain Model gives a better understanding of the Data Dictionary and shows relations between the variables. It is also useful to better understand different levels of abstraction, how to do aggregation of data and the dimensions of the variables. For example, time in phase has the dimension of time units, hours, the value itself and the phase it refers to, for example requirements. We noted that aggregation requires a case by case analysis depending on the nature of the variable (in our case variables that were quotients between different phases could not even be defined at task level) and the level of aggregation at which it will be interpreted. Therefore, the variable also needs to be adequate for the different levels of reporting.

Performance Indicators can be modelled to anticipate quality of implementation or improvement of a process as we demonstrated in building the EEA prediction model. The model is useful to evaluate the quality of implementation of CMMI PP SP1.4 "Estimate Effort and Cost" and, when associated with other performance indicators that influence EEA, can be used to improve it so it becomes more complete. Some of the PIs that were found can be used later in the project to prevent or improve the EEA final value, since EEA as defined in equation 4.6 is a lagging indicator of the performance of the estimation process that still can be improved before the end of the project. The EEA model we built only allows predicting around 30% of the estimation accuracy so to improve results the additional variables that influence EEA should be used. We can follow a similar process to create models and select performance indicators to evaluate other CMMI practices or any process. For that, it is necessary to identify the variables that are inputs and outputs of the process, and understand which equations may allow to evaluate the goal of the process. In our case, the purpose of the process is to be able to have a plan for the development of the software process, but in other cases can be to define the product's requirements, for example. In such situation, in order to evaluate the process, we must understand if the requirements are complete, and compare them to the final requirements and changes introduced during the development process, also considering the quality of those requirements. To model the process it is necessary to build a regression model, based on the organisations process/methods.

6.3 Answering Research Questions

In the following paragraphs we revisit the research questions addressed in this PhD work:

RQ 1- Why do some organisations not achieve the expected benefits when implementing CMMI?

Depending on the methods that the organisation selected to implement the practices, their results may not be those expected. The problems found in organisations implementing CMMI show that if they are not detected and overcome, the implementation can be flawed and they will not perform at the level they aspired to. Part of the problems found are rooted on a poor implementation of MA, a level 2 practice. We discussed those problems in 3.1.3 Problems in Process Improvements, Metrics Programs and CMMI, 4.4.2 CMMI Implementation: Problems and Recommendations, 3.2 Survey on MA Performance in HML Organisations and 5.1.3 Problems Analysis and Limits to Generalisation.

RQ 2- Why does SCAMPI not detect implementation problems, or does not address performance evaluation in all maturity levels?

SCAMPI has still limitations in its sampling and coverage rules and its purpose is not to evaluate performance, as discussed in 3.1.4 SCAMPI Limitations.

RQ 3- What additional recommendations can we provide to organisations to help them avoid problems when implementing CMMI?

We present recommendations in 4.4.2 CMMI Implementation: Problems and Recommendations to help organisations implement CMMI, avoiding the problems found when implementing it. They are compiled on a checklist (see Table 4.1). As part of the implementation problems is rooted on MA and the most challenging levels to implement are the high maturity ones, we included recommendations based on 3.2 Survey on MA Performance in HML Organisations and 5.1.1 Further analysis of the HML Survey Data. The answer regarding CMMI implementation is completed in 5.1.2 Case Studies. The principles presented in 4.5 Manage Configurations Module are relevant to avoid problems and achieve better implementations. Organisations should follow steps aligned with the scientific method, when doing process improvements, which are presented in 4.4.3 Process Improvements, and complemented in 5.2.1 Experiments with Students and 5.2.2 Adoption by an Organisation.

RQ 4- How can we evaluate the quality of implementation of the CMMI practices, ensuring that organisations fully attain their benefits and perform as expected?

We propose that the quality of implementation is measured using performance indicators that measure the results of the practice and its efficiency, effectiveness, and compliance. The EQualPI evaluation procedures exemplify how this is done, in 4.4.1 EQualPI Setup, Tailoring and Evaluation. The management of different evaluations and ability to ensure that process improvements do not result in performance degradation is facilitated following 4.5 Manage Configurations Module. The procedure 4.4.3 Process Improvements allows to ensure that with the process improvement the organisation actually achieves better results and is aligned with the organisation's goals. 5.1.1 Further analysis of the HML Survey Data provides additional evidence to look for, when appraising CMMI organisations.

RQ 5- Is it possible to define metrics to evaluate the quality of implementation of CMMI practices focused on their effectiveness, efficiency and compliance?

4.6.3 Performance Indicator Models: Effort Estimation Evaluation presents the principles of how to define those metrics. To demonstrate that it is possible to evaluate the quality of implementation of a practice focusing on its results, we built the Effort Estimation Accuracy model to evaluate PP SP1.4. The objective of the practice is to provide estimates to plan project execution. We evaluated the effectiveness of the practice through the deviation of the actual effort relative to the estimate, showing how reliable the estimate was. The execution of the project also influences this result, but knowing the percentage it represents on the indicator, as part of the unmeasured variables and uncontrollable factors, we determined how effective the estimate was. The variables, based on following TSP guidelines that we used to build the model, are compliance guidelines. In the particular case of our dataset, the variables were not significant enough to include in the model, but in other processes compliance variables may be relevant. Moreover, the recommended time in phase of DLD Inspections and following the ratio of time spent in Requirements Inspections and Requirements were significant variables influencing the Total EEA (table 5.17). To evaluate the efficiency of the estimation process the time spent planning would be the variable to consider in terms of whether it is worth of closer attention, however we did not have enough data to be able to analyse it. The answer to this question is yes, as we built a model to evaluate the quality of implementation of CMMI PP SP1.4 in section 5.3 Evaluation of the Estimation Process, but there is still further research needed in order to be able to have a more complete evaluation.

RQ 6- Can we determine the effects, expressed in a percentage, of uncontrollable factors in an evaluation metric?

In the EEA model, which is useful to anticipate the quality of the estimates and act on the model coefficients to improve the estimates, the percentage of the controllable factors is given by the model Adjusted R Square, as it represents the percentage of the dependent variable that is explained by our data. That means for the used data we could measure, considering the variables we had access to, the controllable factors represent 30% of the variation. We were not able to identify the uncontrollable factors of these projects but their contribution is reflected in the remainder 70% along with the estimation error and other unmeasured variables. This research question needs further work, to include additional controllable factors we could not consider in the model.

We showed the relationship between the quality of implementation of a CMMI practice and the quality of the outcome of the application of that practice, exemplified on PP SP1.4. Therefore, "it is possible to objectively measure the quality of implementation of the CMMI practices by applying statistical methods, in the analysis of organisations' data, in order to evaluate process improvement initiatives and predict their impact on organisational performance".

6.4 Research Objectives Achievement

Recalling the research objectives:

Objective 1 – Identify problems and difficulties in implementation of CMMI to help define the problem to tackle: considering the high variability of results that CMMI organisations present, evaluate the quality of implementation of practices based on quantitative methods.

Objective 2 – Develop and validate a framework to evaluate the quality of implementation of the CMMI practices.

Objective 3 – Demonstrate the evaluation of quality of implementation by building the performance indicator model to evaluate the particular case of the Project Planning process's Specific Practice 1.4 "Estimate Effort and Cost".

We were able to achieve **Objective 1** and not only defined the problem, but also part of the solution. Our initial results were the base to define and develop the EQualPI framework, its structure, how it is implemented and used.

Regarding the research **Objective 2** we cannot consider it was fully attained. Indeed we developed the Framework and validated the components indicated in 6.2 Validation Discussion, but we were not able to implement it in an organisation and evaluate its processes. Our demonstration was done within the limits we had of data availability. We consider the objective was partially achieved.

We demonstrated EQualPI in the evaluation of the PP SP1.4 "Estimate Effort" as we considered that cost was already affected by the effort parcel. We did achieve the objective of building the model to evaluate the performance of the estimation process, defined variables to build the model and tested part of the hypotheses we considered relevant to validate the Framework. Again, the difficulties in having a dataset with all the necessary data to fully build the model and test all hypotheses were a constraint to completely accomplish research **Objective 3**.

6.5 Challenges and Limits to Generalisation

EQualPI was designed to be aligned with CMMI, it includes a regression model for one of its practices and its procedures are targeting CMMI. Considering that it is based on the problems of metrics programs and other process improvements, it can be used to evaluate other practices of other standards and models. Any set of methods can be mapped with the processes defined to comply with a different model and the measures are defined based on the goal to achieve, rather than the ones in CMMI.

The Domain Model is applicable to iterative and non-iterative development cycles, considering that when there are no cycles the project only executes one. Even though the design of the model was based on the TSP data, the variables used are common in software development, thus applicable to other development processes. The adaptation to projects following Scrum, for instance, would be more challenging even though in some cases it would just require relabelling the variable.

EQualPI is defined conceptually, at high level architecture, and its Metamodel comprises the Repository, its relation with the CMMI framework and the Evaluation. The Procedures package includes the EQualPI Setup, Tailoring, Evaluation, CMMI Implementation and Process Improvement and the module Manage Configurations is defined as well. The EQualPI's Repository package includes the Data Dictionary and Domain Model, while the Performance Indicators Models already has the EEA model. In our research we validated the CMMI Improvement procedure through the analysis of organisations data in the conducted case studies and the analysis of the data of HML organisations surveyed by the SEI; we partially validated the Process Improvements steps in an experiment with graduate and undergraduate students, and such an improvement was adopted by an organisation. The Data Dictionary, we used to extract the data to build a PI model; the Domain Model supported the aggregation of data at different levels of granularity to built the EEA model, and we identified additional PIs to evaluate the Total EEA. We set the stepping stone for EQualPI and research to be continued in order to be able to evaluate processes based on the quality of their implementation. The Framework needs to be extended to other practices and fully populated with corresponding methods, performance indicators and goals. Additionally, it was not instantiated in an organisation, which may be challenging for its complexity. In any case, to gather the data it is still necessary that the organisation collects them, with all the inherent challenges that it presents.

We identified several factors that impact on the generalisation of the research results and, awareness of that, allowed us to keep them controlled to isolate their interference in the results, as best we could. We kept a record of context and those factors that are not object of our research, to allow repeatability. These factors are:

Human Factors

The experience of a project team influences the results of projects. Also the quality of individuals' work can positively or negatively influence the project's results. The quality of the data used to build the models depends on the people's rigour recording it. The Benford statistic helped us ensure that the effort data in the TSP repository was reliable.

Aggregation Factors

When aggregating results of individuals and different teams, inaccurate data points' noise can be cancelled by others. In other situations, the outputs of the team influences the following phase. Such noise can change the aggregated information. Once we detected inaccurate data, e.g. several projects with same defects descriptions, we isolated them, not considering them in the model.

Complexity Factors

The complexity of projects depends on different factors such as size, duration, complexity of the product, newness of the technology, etc. We also recorded other context factors such as team size and distribution. Those factors were documented but not included in the model as they are part of the error.

Biasing Factors

Other factors that limit the generalisation of the research results are the bias of the researcher and the analysed organisations. In general, to avoid researcher's bias the work was submitted to other researchers' reviews, conferences and journals. Organisations themselves can bias the results by withholding certain information or altering the data shared. For this reason in the data analysis we analysed data of several organisations, when possible. We also did sanity checks on the received organisations' data.

Measurement Selection Bias

Grimstad and Jorgensen (2006) highlight three measurement selection factors that are particularly important:

1. Exclusion of cancelled projects, leading to a too positive view of estimation;

2. Exclusion of estimated projects that never started, leading to a negative view of estimation (according to the authors, optimistic plans would be more likely to win bidding, for example);

3. Inclusion of projects to "confirm the desired output of the analysis" while omitting others, named "confirmation" bias.

We did not consider cancelled projects before start, as they would not be present in the database; we also did not find cases of projects that were estimated but did not have any records of data resultant from their execution. These two biasing factors were out of the scope of the research. Even though we did not find cases of cancellation before even starting we only considered completed parts in the design of the model, ignoring the parts that were planned but not executed.

The decision was based on the fact that we had no absolute confirmation of whether the workbook was of a completed project or referring to an intermediate development cycle.

Regarding the third factor we did not select projects by convenience but we also did not get the degree of variety we desired as had we been able to get the same data from other repositories, since we just evaluated the effects of using methods that are common to TSP.

6.6 Future Research Work

As future research work, it is necessary to extend the Framework to other practices and methods, also including more performance models, and to instantiate the EEA model in an organisation. Building the models needs to be done in the organisation environment and also calibrating them to their data, after the first processes cycles. The following steps are necessary:

- Identify the model to implement (for example, CMMI or a subset, TSP, Scrum);
- Follow EQualPi's setup procedures;
- Identify the practices to evaluate;
- Check and review existing PIs;
- Add missing PIs as needed;
- Build and calibrate the leading PIs models;
- Evaluate the quality of implementation considering the organisation goals;
- Follow improvement steps to evolve processes and allow achieving performance goals.

Implementing EQualPI in organisations will allow to have data of different maturity levels. The diversity of data will make it possible to map maturity profiles to other methods and performance indicators. In the future, EQualPI can be used for benchmarking, supporting multiple clients who eventually will be able to compare their performance with the anonymous data of others who allow selected data of theirs to be used by others. The Data Dictionary is already prepared to support multiple organisations, but the architecture of the Framework needs an additional package in the business layer to perform data consistency checks.

The implementation checklist of problems and recommendations should be used in an organisation implementing CMMI, that would follow the guidelines. As a result the quality and usefulness would be assessed in the organisation and the new lessons learnt would improve the checklist. Conducting similar case studies to the ones we did can also enrich the CMMI Implementation procedures. We recommend the use of the design used in CI but would warn that researchers may find difficulties in conducting a thorough case study, and may face trust issues, when accessing documentation or interviewing people.

The Domain Model should evolve to be useful to different types of development cycles. The aggregation rules need to be defined case by case, depending on the variable, as we are concious they are not a mere sum of values.

With a more complete dataset, additional context information and even other Process Variables, we can improve the EEA model. Some variables such as CMMI, time in Requirements and High Level Design phases found little representation in our dataset. The initial questionnaires that are given to TSP teams are important sources of information about the product and nature of the project, such as complexity and programming language, and team experience. We had few data points, but having that information for all projects could help improve the EEA model. Caution must be exercised when adding variables to PIs models as the sample size will also have to increase if the number of variables significantly increases, to respect the recommendation given when building the model of the ratio between independent variables and number of subjects. We built the EEA model using linear regression; other techniques to build it should result in more accurate results, we would only recommend that for such models to be useful they should have instructions of how to use them, including of how to act on the variables. We also consider that the methodology used to build the EEA model should be tested with other estimation methods so that the differences between methods can be compared. For example, building an EEA model of the Scrum planning poker and project management methodology would require variables such as user stories with their size measured in story points, mapped with the corresponding tasks to implement them and collecting the time estimated in each task and time actually spent on it, and also analyse the scope (user stories) completed by the end of each sprint.

Another research direction to explore is to determine the influences of partial estimates and effects of re-estimation, through analysing projects with several cycles and re-launch meetings, and comparing the sum of the partial estimates, with a global estimate given at the beginning of the project and the final global estimate.

EQualPI's procedure for Process Improvements was only partially validated, therefore should be reproduced in an industrial setting. The pilot should have a control group; normally comparing with the baseline could be sufficient but in some cases it is not, for instance, when wanting to compare different solutions. Even if repeating our experiment with a group of students there should be a group using another classifiers list, to be able to measure the improvement effects better.

In conclusion, our main contributions to the science and industry are the EQualPI's principles, metamodel, procedures and repository, providing:

- Metamodel and alignment of the Framework with CMMI (or other improvement models), practices and goals;
- Data Dictionary and Domain Model, to document and understand the relations between the variables to be measured in an organisation;
- Factors, problems and recommendations for organisations implementing CMMI, which can be used to evaluate and improve performance;

- Set of PIs and the EEA model, to evaluate CMMI PP SP1.4

We also contribute with the methodology we used to implement and validate our PIs and the methodology we used to implement EQualPI that shall be used to extend the framework. This PhD is an important contribution to the Software Engineering and Process Improvement areas of research. The CMMI Institute can benefit from EQualPI to improve appraisals and provide tools for the community. They revealed they were interested in the Framwework in our last participation in SEPG Europe 2013. The CMMI community also responded positively to a survey we conducted regarding EQualPI and the relevance of the requirements we specified for the Framework and its purposes (see D.2 Results in appendix D Survey About EQualPI).

EQualPI goes beyond evaluating maturity or capability, as CMMI does, in order to evaluate the quality of implemented processes through their outcomes and analysing their performance. We recognise it is a challenge to implement. Using it requires a good understanding of the process improvement model to use (CMMI or other) as well as understanding the EQualPI framework. It has an additional statistical complexity and it is a challenge to design models that are used in processes executed by humans and that involve creativity, adding subjective factors to the set of variables influencing the models, such as: motivation, fear, tiredness, instinct/intuition, experience, empathy, concern, stress, and many other. Setting up the EQualPI framework in an organisation also has all the challenges of setting up a metrics program or implementing CMMI HML. It comes with the complexity of establishing and understanding the relations between practices and PIs, respectively, and amongst them. There is still a lot of work to do to extend the Framework and establish those relations but the groundwork and initial validation are done.

Organisations implementing CMMI just for compliance might as well ensure they achieve the performance benefits of implementing the model. EQualPI differs from CMMI in recognising there are differences of performance that reside in the quality of implementation. Those differences occur and matter beyond just going up a capability or maturity level.

References

Abreu, Fernando Manuel Pereira da Costa Brito e. *Engenharia de software orientado a objectos: uma aproximação quantitativa*. Doctoral thesis, Universidade Técnica de Lisboa. Instituto Superior Técnico, 2001.

Ackerman, A. Frank, Lynne S. Buchwald, and Frank H. Lewski. Software Inspections: An Effective Verification Process. *IEEE Software*, 6(3):31–36, 1989.

Alaa, F. S. and A. Al-Afeef. A GP effort estimation model utilizing line of code and methodology for NASA software projects. In *Intelligent Systems Design and Applications (ISDA), 2010 10th International Conference on*, pages 290–295, 2010.

Albrecht, A. J. and Jr. Gaffney, J. E. Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation. *Software Engineering, IEEE Transactions on*, SE-9(6):639–648, 1983.

Álvarez, José M, Andy Evans, and Paul Sammut. Mapping between Levels in the Metamodel Architecture. In Gogolla, Martin and Cris Kobryn, editors, *UML 2001 – The Unified Modeling Language. Modeling Languages, Concepts, and Tools*, volume 2185 of *Lecture Notes in Computer Science*, pages 34–46. Springer Berlin Heidelberg, 2001.

Anonymous Research Group, . Search nach Measures in CMMI V1.2, April 2007. Unpublished work.

Apfelbaum, Larry and John Doyle. Model based testing. In *10th International Software Quality Week Conference*, San Francisco, 1997.

Armstrong, James, Richard Barbour, Richard Hefner, and David H. Kitson. Standard CMMI^{SM} Appraisal Method for Process Improvement (SCAMPI^{SM}): Improvements and Integration. *Systems Engineering*, 5(1):19–26, 2002.

Bailey, John W. and Victor R. Basili. A meta-model for software development resource expenditures. In *Proceedings of the 5th International Conference on Software Engineering*, pages 107–116, San Diego, California, United States, 1981. IEEE Press.

Barcellos, Monalessa Perini. *Uma Estratégia para Medição de Software e Avaliação de Bases de Medidas para Controlo Estatístico de Processos de Software em Organizações de Alta Maturidade*. Doctoral, Universidade Federal do Rio de Janeiro, 2009.

Basili, Victor R. and David M. Weiss. Evaluation of a software requirements document by analysis of change data. In *Proceedings of the 5th International Conference on Software Engineering*, pages 314–323, San Diego, California, United States, 1981. IEEE Press.

Bell, T. E. and T. A. Thayer. Software requirements: Are they really a problem? In *Proceedings of the 2nd International Conference on Software Engineering*, pages 61–68, San Francisco, California, United States, 1976. IEEE Computer Society Press.

Blackburn, J. D., G. D. Scudder, and L. N. Van Wassenhove. Improving speed and productivity of software development: a global survey of software developers. *Software Engineering, IEEE Transactions on*, 22(12):875–885, 1996.

Boehm, Barry W. Software Engineering Economics. Prentice Hall Inc., New Jersey, 1981.

Bollinger, Terry and Clement McGowan. A Critical Look at Software Capability Evaluations: An Update. *IEEE Software*, 26(5):80–83, 2009.

Boria, J. L., 2007. Personal Communication.

Braga, Petrônio L., Adriano L. I. Oliveira, and Silvio R. L. Meira. A GA-based feature selection and parameters optimization for support vector regression applied to software effort estimation. In *Proceedings of the 2008 ACM Symposium on Applied Computing*, pages 1788–1792, Fortaleza, Ceara, Brazil, 2008. ACM.

Breuker, Dennis, Jacob Brunekreef, Jan Derriks, and Ahmed Nait Aicha. Reliability of software metrics tools. In *International Conference on Software Process and Product Measurement*, pages 10–20, Amsterdam, 2009.

Brooks, F. *The Mythical Man Month: Essays on Software Engineering*. Addison Wesley, Prentice-Hall, London, 1982.

Buehler, R., D. Griffin, and M. Ross. Exploring the 'Planning Fallacy': why people underestimate their task completion times. *Journal of Personality and Social Psychology*, 67(3):366–381, 1994.

Byrnes, Paul D. What's All the Fuss. . . What's Really Different About SCAMPI V1.3. In *SEPG Europe*, Dublin, Ireland, 2011. CMU/SEI.

Callison, Rachel and Marlene MacDonald. A Bibliography of the Personal Software Process (PSP) and the Team Software Process (TSP). Technical Report CMU/SEI-2009-SR-025, CMU/SEI, October 2009 2009.

Campo, Michael. Why CMMI Maturity Level 5? *CROSSTALK The Journal of Defense Software Engineering*, (January/February):15–18, 2012.

Canfora, G., Félix García, M. Piattini, F. Ruiz, and C. Visaggio. Applying a framework for the improvement of software process maturity. *Software Practice and Experience*, 36(3):283–304, 2006.

Card, David. Defect Analysis: Basic Techniques for Management and Learning. *Advances in Computers*, 65:259–295, 2005.

Card, David N. Learning from Our Mistakes with Defect Causal Analysis. *IEEE Softw.*, 15(1): 56–63, 1998.

Charette, Robert, Laura M. Dwinnell, and John McGarry. Understanding the Roots of Process Performance Failure. *CROSSTALK The Journal of Defense Software Engineering*, 17(8):18–22, 2004.

Chen, Jie-Cherng and Sun-Jen Huang. An empirical analysis of the impact of software development problem factors on software maintainability. *Journal of Systems and Software*, 82(6): 981–992, 2009.

Chillarege, Ram, Inderpal S. Bhandari, Jarir K. Chaar, Michael J. Halliday, Diane S. Moebus, Bonnie K. Ray, and Man-Yuen Wong. Orthogonal Defect Classification - A Concept for In-Process Measurements. *IEEE Transactions on Software Engineering*, 18(11):943–956, 1992.

Chrissis, Mary Beth, Mike Konrad, and Sandy Shrum. *CMMI for Development®: Guidelines for Process Integration and Product Improvement*. SEI Series in Software Engineering. Addison-Wesley, Massachusetts, 3 edition, 2011.

CISQ, . Automated Function Points (AFP), 2014. URL http://it-cisq.org/wp-content/uploads/2014/11/Automated-Function-Points-Specification-OMG-Formal-January-2014.pdf. Last accessed: 11-12-2015.

CISQ, . CISQ Code Quality Standards, 2016. URL http://it-cisq.org/standards/. Last accessed: 04-01-2016.

CMMI Product Team, . CMMI® for Development, Version 1.3. Technical Report CMU/SEI-2010-TR-033, ESC-TR-2010-033, CMU/SEI, November 2010.

CMU/SEI, . News Items - Carnegie Mellon SEI and OMG Announce the Launch of CISQ—The Consortium for IT Software Quality.

CMU/SEI, . Software Engineering Measurement and Analysis Initiative. Technical report, CMU/SEI, 2001.

CMU/SEI, . Criteria for Audits of High Maturity Appraisals, 2008. URL http://www.sei.cmu.edu/cmmi/solutions/appraisals/himataudits.cfm. Last accessed: 21-06-2012.

CMU/SEI, . A Practical Approach for Building CMMI Process Performance Models, 2009. URL http://www.sei.cmu.edu/webinars/view_webinar.cfm?webinarid=18622. Webinar video. Last accessed: 16-10-2016.

CMU/SEI, . CMMI® For Development SCAMPI^{SM} Class A Appraisal Results 2009 End-Year Update. Technical report, March 2010a.

CMU/SEI, . CMMI®For Development SCAMPI^{SM} Class A Appraisal Results 2010 Mid-Year Update. Technical report, September 2010b.

CMU/SEI, . Accelerated Improvement Method (AIM). Technical report, May 2010 2010c.

CMU/SEI, . CMMI® for Development SCAMPI^{SM} Class A Appraisal Results 2010 End-Year Update. Technical report, March 2011a.

CMU/SEI, . CMMI® for SCAMPI^{SM} Class A Appraisal Results 2011 Mid-Year Update. Technical report, September 2011b.

CMU/SEI, . Standard CMMI® Appraisal Method for Process Improvement (SCAMPI^{SM}) A, Version 1.3: Method Definition Document. Technical Report CMU/SEI-2011-HB-001, CMU/SEI, 2011c. SCAMPI Upgrade Team.

CMU/SEI, . CMMI® for SCAMPI^{SM} Class A Appraisal Results 2011 End-Year Update. Technical report, March 2012a.

CMU/SEI, . CMMI® for SCAMPI^{SM} Class A Appraisal Results 2012 Mid-Year Update. Technical report, September 2012b.

Colombo, A., E. Damiani, F. Frati, S. Oltolina, K. Reed, and G. Ruffatti. The Use of a Meta-Model to Support Multi-Project Process Measurement. In *Software Engineering Conference, 2008. APSEC '08. 15th Asia-Pacific*, pages 503–510, 2008.

Consultant, D., 2009. Personal Communication.

Conte, S. D., H. E. Dunsmore, and V. Y. Shen. *Software engineering metrics and models*. Benjamin-Cummings Publishing Co., Inc., 1986.

Curtis, B., D. Reifer, G. V. Seshagiri, I. Hirmanpour, and G. Keeni. The Case for Quantitative Process Management. *IEEE Software*, 25(3):24–28, 2008.

Curtis, Bill. Software Quality Measurement. In *Software Assurance Forum*. CMU/SEI, 2010.

Davis, Noopur and Jim McHale. Relating the Team Software Process (TSP^{SM}) to the Capability Maturity Model® for Software (SW-CMM®). Technical Report CMU/SEI-2002-TR-008, ESC-TR-2002-008, CMU/SEI, March 2003.

DeMarco, Tom. *Controlling Software Projects Management Measurement & Estimation*. Yourdon Press a Prentice-Hall Company, Englewood Cliffs, 1982.

Diaz, Michael and Joseph Sligo. How Software Process Improvement Helped Motorola. *IEEE Software*, pages 75–81, 1997.

Dybå, Tore. *Experiences in Process Modelling and Enactment: an Investigation of the Importance of Organisational Issues*. Doctoral dissertation, Norwegian University of Science and Technology, 2001.

Faria, Pascoal. A Path for Performance Improvement: the Personal Software Process (PSP) and the Team Software Process (TSP), 6-11-2009 2009. URL `https://paginas.fe.up.pt/~prodei/dsie09/docs/joaopascoalfaria/PSP_TSP_DSIE_05Fev09_Final.pdf`. Last accessed 05-06-2010, prior version in the given address.

Fenton, N. E. and M. Neil. Software Metrics: Success, Failures and New Directions. *Journal of Systems and Software*, 47:149–157, 1999.

Ferguson, Bob. Leading Indicators of Program Management. In *8th Annual CMMI Technology Conference & User Group*, Denver, Colorado, 2008. Defense Technical Publication Center.

Ferguson, Robert, Dennis Goldenson, James McCurley, Robert Stoddard, David Zubrow, and Debra Anderson. Quantifying Uncertainty in Early Lifecycle Cost Estimation (QUELCE). Technical Report CMU/SEI-2011-TR-02, 2011.

Finnie, Gavin R., Gerhard E. Wittig, and Doncho I. Petkov. Prioritizing software development productivity factors using the analytic hierarchy process. *J. Syst. Softw.*, 22(2):129–139, 1993.

Fleiss, Joseph L. Measuring Nominal Scale Agreement Among Many Raters. *Psychological Bulletin*, 76(5):378–382, 1971.

Florac, William A., Anita D. Carleton, and Julie R. Barnard. Statistical Process Control: Analyzing a Space Shuttle Onboard Software Process. *IEEE Softw.*, 17(4):97–106, 2000.

Foss, T., E. Stensrud, B. Kitchenham, and I. Myrtveit. A simulation study of the model evaluation criterion MMRE. *Software Engineering, IEEE Transactions on*, 29(11):985–995, 2003.

Freimut, Bernd, Christian Denger, and Markus Ketterer. An Industrial Case Study of Implementing and Validating Defect Classification for Process Improvement and Quality Management. In *Proceedings of the 11th IEEE International Software Metrics Symposium*, page 19. IEEE Computer Society, 2005.

Fuggetta, Alfonso. Software process: a roadmap. In *Proceedings of the Conference on The Future of Software Engineering*, pages 25–34, Limerick, Ireland, 2000. ACM.

Fulton, Gregory P. SEI CMM Level 5: Lightning Strikes Twice. *CROSSTALK The Journal of Defense Software Engineering*, 15(9):22–24, 2002.

García, F., M. Bertoa, C. Calero, A. Vallecillo, and F. Ruiz. Towards a consistent terminology for software measurement. *Information and Software Technology*, 48(8):631–644, 2006.

García, F., M. Serrano, J. Cruz-Lemus, F. Ruiz, and M. Piattini. Managing software process measurement: A metamodel-based approach. *Information Sciences*, 177(12):2570–2586, 2007.

García, Félix, Francisco Ruiz, José Cruz, and Mario Piattini. Integrated Measurement for the Evaluation and Improvement of Software Processes. volume 2786 of *Lecture Notes in Computer Science*, pages 94–111. Springer Berlin / Heidelberg, 2003.

Goh, T. N., M. Xie, and W. Xie. Prioritizing Process in Initial Implementation of Statistical Process Control. *IEEE Transactions on Engineering Management*, 45(1):66–72, 1998.

Goldenson, Dennis R., Diane L. Gibson, and Robert W. Ferguson. Why Make the Switch? Evidence about the Benefits of CMMI. In *SEPG*. CMU/SEI, 2004.

Goldenson, Dennis R., James McCurley, and Robert W. Stoddard II. Use and Organizational Effects of Measurement and Analysis in High Maturity Organizations: Results from the 2008 SEI State of Measurement and Analysis Practice Surveys. Technical report, CMU/SEI, 2008.

Gopal, A., M. S. Krishnan, T. Mukhopadhyay, and D. R. Goldenson. Measurement programs in software development: determinants of success. *Software Engineering, IEEE Transactions on*, 28(9):863–875, 2002.

Gou, L., Q. Wang, J. Yuan, Y. Yang, M. Li, and N. Jiang. Quantitative Defects Management in Interative Development with BiDefect. *Software Process Improvement and Practice*, 14(4): 227–241, 2009.

Goulão, Miguel Carlos Pacheco Afonso. *Component-Based Software Engineering: a Quantitative Approach*. Doctoral, Universidade Nova de Lisboa, Faculdade de Ciências e Tecnologia, 2008.

Grady, Robert B. *Practical software metrics for project management and process improvement*. Prentice-Hall, Inc., 1992.

Gremba, Jennifer and Chuck Myers. The IDEAL *SM* Model: A Practical Guide for Improvement. *Bridge*, (3), 1997. URL http://www.sei.cmu.edu/library/assets/idealmodel.pdf. Last accessed: 05-10-2016.

Grimstad, S., M. Jorgensen, and K. Molokken-Ostvold. The clients' impact on effort estimation accuracy in software development projects. In *Software Metrics, 2005. 11th IEEE International Symposium*, pages 10 pp.–10, 2005.

Grimstad, Stein and Magne Jorgensen. A framework for the analysis of software cost estimation accuracy. In *Proceedings of the 2006 ACM/IEEE international symposium on Empirical software engineering*, pages 58–65, Rio de Janeiro, Brazil, 2006. ACM.

Hahn, Gerald J., William J. Hill, Roger W. Hoerl, and Stephen A. Zingraph. The Impact of Six Sigma Improvement - A Glimpse into the Future of Statistics. *The American Statistician*, 53 (3):208–215, 1999.

Halkjelsvik, T. and M. Jørgensen. From Origami to Software Development: A Review of Studies on Judgment-Based Predictions of Performance Time. *Psychological Bulletin*, 138(2):238–271, 2012.

Hamill, Maggie and Goseva-Popstojanova Katerina. Common Trends in Software Fault and Failure Data. *IEEE Trans. Softw. Eng.*, 35(4):484–496, 2009.

Hamon, Patrick and Olivier Pinette. Les indicateurs Mesure & Analyse. Technical report, Spirula, 22-06-2010 2010. Mauvaises pratiques.

Hari, CH. V. M. K. and P. V. G. D. Prasad Reddy. A Fine Parameter Tuning for COCOMO 81 Software Effort Estimation using Particle Swarm Optimization. *Journal of Software Engineering*, 5 (1):38–48, 2011.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer, 2 edition, 2009.

Hayes, J. Huffman. Building a Requirement Fault Taxonomy: Experiences from a NASA Verification and Validation Research Project. In *Proceedings of the International Symposium on Software Reliability Engineering*, pages 49 – 59, Denver, CO, 2003. IEEE Computer Society.

Hayes, Jane Huffman, Inies Raphael, David M. Pruett, and Elizabeth Ashlee Holbrook. Case History of International Space Station Requirement Faults. In *Proceedings of the 11th IEEE International Conference on Engineering of Complex Computer Systems*, pages 17–26, Standford, California, 2006. IEEE Computer Society.

Hefner, Richard. The True Costs and Benefits of CMMI Level 5. In *Systems and Software Technology Conference*, 2009.

Henningsson, Kennet and Claes Wohlin. Assuring Fault Classification Agreement - An Empirical Evaluation. In *International Symposium on Empirical Software Engineering*, pages 95 –104, Redondo Beach, California, 2004. IEEE Computer Society.

Herbsleb, J. D. and D. R. Goldenson. A systematic survey of CMM experience and results. In *Proceedings of the 18th International Conference on Software Engineering (ICSE)*, pages 323–330, 1996.

Hihn, J. and H. Habib-agahi. Cost estimation of software intensive projects: a survey of current practices. In *Software Engineering, 1991. Proceedings., 13th International Conference on*, pages 276–287, 1991.

Hill, J., L. C. Thomas, and D. E. Allen. Experts' estimates of task durations in software development projects. *International Journal of Project Management*, 18:13–21, 2000.

Hollenbach, Craig and Doug Smith. A portrait of a CMMI^{SM} level 4 effort . *Systems Engineering*, 5(1):52–61, 2002.

Hsueh, Nien-Lin, Wen-Hsiang Shen, Zhi-Wei Yang, and Don-Lin Yang. Applying UML and software simulation for process definition, verification, and validation. *Inf. Softw. Technol.*, 50 (9-10):897–911, 2008.

Humphrey, Watts S. The Software Engineering Process: Definition and Scope. In *4th Int Software Process Workshop*, pages 82–83, 1988.

Humphrey, Watts S. Introduction to Software Process Improvement. Technical Report CMU/SEI-92-TR-7, CMU/SEI, June 1992 (Revised June 1993) 1992.

Humphrey, Watts S. *PSP^{SM} A Self-Improvement Process for Software Engineers*. SEI Series in Software Engineering. Addison-Wesley, 2005.

Humphrey, Watts S. *TSP^{SM}: Coaching Development Teams*. The SEI Series in Software Engineering. Addison-Wesley, 2006.

IEEE Std 1044-1993. IEEE Standard Classification for Software Anomalies. Standard, IEEE, 1993.

IEEE Std 1044-2009. IEEE Standard Classification for Software Anomalies. Standard, IEEE, 2009.

IEEE Std 610:1990. IEEE Standard Glossary of Software Engineering Terminology. Standard, IEEE, 1990.

IEEE Std 830-1998. IEEE Recommended Practice for Software Requirements Specifications. Standard, IEEE, 1998.

Investopedia, . What are leading, lagging and coincident indicators? What are they for?, 2007. URL http://www.investopedia.com/ask/answers/177.asp. Last accessed 30-05-2011.

ISO 14598:1998. Information Technology - Software Product Evaluation. Standard, ISO, 1998.

ISO/FDIS 9126-1:2000. FDIS 9126-1 Software Engineering - Product quality - Part 1: Quality model. Standard, International Organization for Standardization, 2001.

ISO/IEC 15939:2007. Systems and software engineering – Measurement process. Standard, ISO, 2008.

Jeffery, R. and M. Berry. A framework for evaluation and prediction of metrics program success. In *Software Metrics Symposium, 1993. Proceedings., First International*, pages 28–39, 1993.

Johnson, Philip M., Hongbing Kou, Michael Paulding, Qin Zhang, Aaron Kagawa, and Takuya Yamashita. Improving Software Development Management through Software Project Telemetry. *IEEE Softw.*, 22(4):76–85, 2005.

Jones, Capers. *Applied Software Management: Assuring Productivity and Quality*. Software Engineering Series. McGraw-Hill, Inc., New York, 1991.

Jones, Capers. *Software Engineering Best Practices - Lessons from Successful Projects in the Top Companies*. McGraw Hill, 2010.

Jørgensen, Magne. Regression Models of Software Development Effort Estimation Accuracy and Bias. *Empirical Software Engineering*, 9(4):297–314, 2004.

Jørgensen, Magne. A Critique of How We Measure and Interpret the Accuracy of Software Development Effort Estimation. In *1st International Workshop on Software Productivity Analysis and Cost Estimation*, pages 15–22, 2007.

Jørgensen, Magne and Martin Shepperd. A Systematic Review of Software Development Cost Estimation Studies. *Software Engineering, IEEE Transactions on*, 33(1):33–53, 2007.

Jun, Eung Sup and Jae Kyu Lee. Quasi-optimal case-selective neural network model for software effort estimation. *Expert Systems with Applications*, 21(1):1–14, 2001.

Juran, Joseph M. and A. Blanton Godfrey. *Juran's Quality Handbook*. McGraw-Hill, New York, 1998.

Kalinowski, Marcos, Emilia Mendes, David Card, and Guilherme Travassos. Applying DPPI: A Defect Causal Analysis Approach Using Bayesian Networks. In Ali Babar, M., Matias Vierimaa, and Markku Oivo, editors, *Product-Focused Software Process Improvement*, volume 6156 of *Lecture Notes in Computer Science*, pages 92–106. Springer Berlin / Heidelberg, 2010.

Kasunic, Mark. Performance Benchmarking Consortium. In *6th Annual CMMI Technology Conference & User Group*, Denver, Colorado, 2006. National Defense Industrial Association, Defense Technical Information Center.

Kasunic, Mark. A Data Specification for Software Project Performance Measures: Results of a Collaboration on Performance Measurement. Technical Report CMU/SEI-2008-TR-012 ESC-TR-2008-012, 2008.

Kitchenham, Barbara. Guidelines for performing Systematic Literature Reviews in Software Engineering. Joint Report EBSE-2007-01, Keele University and University of Durham, 9 July 2007.

Kitchenham, Barbara, Shari Lawrence Pfleeger, and Norman Fenton. Towards a Framework for Software Measurement Validation. *IEEE Trans. Softw. Eng.*, 21(12):929–944, 1995.

Kitchenham, Barbara, Cat Kutay, Ross Jeffrey, and Colin Connaughton. Lessons Learnt from the Analysis of Large-scale Corporate Databases. In *International Conference on Software Engineering*, pages 439 – 444, Shanghai, 2006. ACM.

Knecht, William. Pilot Willingness to Take Off Into Marginal Weather, Part II: Antecedent Overfitting With Forward Stepwise Logistic Regression. Technical Report DOT/FAA/AM-05/15, Civil Aerospace Medical Institute, 2005.

Knuth, Donald E. An Emprirical Study of FORTRAN Programs. *Software - Practice and Experience*, 1:105–133, 1971.

Kocaguneli, Ekrem and Tim Menzies. How to Find Relevant Data for Effort Estimation? In *Empirical Software Engineering and Measurement (ESEM), 2011 International Symposium on*, pages 255–264, 2011.

Kueng, Peter. Process performance measurement system: a tool to support process-based organizations. *Total Quality Management*, 11(1):67 – 85, 2000.

Kumar, J. N. V. R. S., T. G. Rao, Y. N. Babu, S. Chaitanya, and K. Subrahmanyam. A novel method for software effort estimation using inverse regression as firing interval in fuzzy logic. In *Electronics Computer Technology (ICECT), 2011 3rd International Conference on*, volume 4, pages 177–182, 2011.

Landis, J. R. and G. G. Koch. The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174, 1977.

Lederer, A. L., R. Mirani, B.S. Neo, J. Pollard, Prasad, C., and K. Ramamurthy. Information system and estimating: a management perspective. *MIS Quarterly*, 14(2):159–176, 1990.

Leek, Jeff. Predictions Study Design. Technical report, Johns Hopkins Bloomerang School, 2013.

Leeson, Peter. Why the CMMI® does not work. In *SEPG Europe*, Prague, Czech Republic, 2009. CMU/SEI.

Liebchen, Gernot A. and Martin Shepperd. Data sets and data quality in software engineering. In *Proceedings of the 4th international workshop on Predictor models in software engineering*, pages 39–44, Leipzig, Germany, 2008. ACM.

Lokan, C. and E. Mendes. Applying moving windows to software effort estimation. In *Empirical Software Engineering and Measurement, 2009. ESEM 2009. 3rd International Symposium on*, pages 111–122, 2009.

Lopes Margarido, Isabel. Requirements Defects Classification List. Technical Report PRODEI-0903-TR-001, Faculty of Engineering, University of Porto, August 2010. URL http://paginas.fe.up.pt/~pro09003/publications.html. Last accessed: 06-11-2016.

Lopes Margarido, Isabel. Summary of Literature Review on Effort Estimation. Technical Report PRODEI-0903-TR-003, Faculty of Engineering, University of Porto, 2012a. URL https://paginas.fe.up.pt/~pro09003/juridocs.html. Last accessed: 06-11-2016.

Lopes Margarido, Isabel. Thesis Documentation: Data Dictionary, 2012b. URL https://paginas.fe.up.pt/~pro09003/juridocs.html. Last accessed: 05-11-2016.

Lopes Margarido, Isabel. Case Study: Software Engineering Excellence - From Good to Great, 2013. URL https://paginas.fe.up.pt/~pro09003/wordpress.html#linkcs. Last accessed: 31-12-2015.

Lopes Margarido, Isabel, João Pascoal Faria, Marco Vieira, and Raul Moreira Vidal. Classification of Defect Types in Requirements Specifications: Literature Review, Proposal and Assessment. In *Information Systems and Technologies (CISTI), 2011 6th Iberian Conference on*, pages 555–561, Chaves, Portugal, 2011a. IEEE.

Lopes Margarido, Isabel, João Pascoal Faria, Marco Vieira, and Raul Moreira Vidal. CMMI Practices: Evaluating the Quality of the Implementation. In *SEPG Europe*, Dublin, Ireland, 2011b. CMU/SEI.

Lopes Margarido, Isabel, João Pascoal Faria, Raul Moreira Vidal, and Marco Vieira. *Towards a Framework to Evaluate and Improve the Quality of Implementation of CMMI® Practices*, volume 7343 of *Product-Focused Software Development and Process Improvement*. Springer Berlin / Heidelberg, Madrid, 2012.

Lopes Margarido, Isabel, João Pascoal Faria, Raul Moreira Vidal, and Marco Vieira. Challenges in Implementing CMMI® High Maturity: Lessons Learned and Recommendations. *Software Quality Professional*, 16(1), 2013.

Lopez-Martin, Cuauhtemoc. A fuzzy logic model for predicting the development effort of short scale programs based upon two independent variables. *Applied Soft Computing*, 11(1):724–732, 2011.

Lutz, Robyn R. and Carmen Mikulski. Requirements discovery during the testing of safety-critical software. In *Proceedings of the 25th International Conference on Software Engineering*, pages 578–583, Portland, Oregon, 2003. IEEE Computer Society.

MacDonell, S. G. and M. J. Shepperd. Comparing Local and Global Software Effort Estimation Models - Reflections on a Systematic Review. In *Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on*, pages 401–409, 2007.

Macleod Clark, J. and L. Hockey. *Research for nursing: a guide for the enquiring nurse*. John Wiley, New York, 1981.

Masters, Steve, PhD Sandi Behrens, Judah Mogilensky, and Charlie Ryan. SCAMPI Lead Appraiser^{SM} Body of Knowledge (SLA BOK). Technical Report CMU/SEI-2007-TR-019, ESC-TR-2007-019, CMU/SEI, 2007.

Maxwell, Katrina, Luk Van Wassenhove, and Soumitra Dutta. Performance Evaluation of General and Company Specific Models in Software Development Effort Estimation. *Manage. Sci.*, 45 (6):787–803, 1999.

McAndrews, Donald R. The Team Software Process (TSP^{SM}): An Overview and Preliminary Results of Using Disciplined Practices. Technical Report CMU/SEI-2000-TR-015, ADA387260)., CMU/SEI, 2000.

McCarthy, Lawrence. Piloting Results-Based Appraisals. In *9th Annual CMMI Technology Conference & User Group*, Denver, Colorado, 2009. National Defense Industrial Association, Defense Technical Information Center.

McCurley, James and Dennis R. Goldenson. Performance Effects of Measurement and Analysis: Perspectives from CMMI High Maturity Organizations and Appraisers. Technical report, CMU/SEI, 2010.

McFeeley, Bob. IDEAL^{SM}: A User's Guide for Software Process Improvement. Technical Report CMU/SEI-96-HB-001, CMU/SEI, 1996.

McGarry, John, David Card, Cheryl Jones, Beth Layman, Elizabeth Clark, Joseph Dean, and Fred Hall. *Practical Software Measurement: Objective Information for Decision Makers*. Addison-Wesley, 2002.

McHale, James, Timothy A. Chick, and Eugene Miluk. Implementation Guidance for the Accelerated Improvement Method (AIM). Technical Report CMU/SEI-2010-SR-032, CMU/SEI, 2010.

Menzies, T., D. Port, Chen Zhihao, and J. Hihn. Validation methods for calibrating software effort models. In *Software Engineering, 2005. ICSE 2005. Proceedings. 27th International Conference on*, pages 587–595, 2005a.

Menzies, Tim, Dan Port, Zhihao Chen, and Jairus Hihn. Specialization and extrapolation of software cost models. In *Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering*, pages 384–387, Long Beach, CA, USA, 2005b. ACM.

Miller, George A. The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. *The Psychological Review*, 63:81–97, 1956.

Mishra, S. and B. H. Schlingloff. Compliance of CMMI Process Area with Specification Based Development. In *Software Engineering Research, Management and Applications, 2008. SERA '08. Sixth International Conference on*, pages 77–84, 2008.

Moen, Ronald and Clifford Norman. Evolution of the PDCA Cycle, 2006. URL pkpinc.com/files/NA01MoenNormanFullpaper.pdf. Last accessed: 05-10-2016.

Mogyorodi, Gary. Requirements-based testing: an overview. In *39th International Conference and Exhibition on Technology of Object-Oriented Languages and Systems (TOOLS39)*.

Mohagheghi, Parastoo, Bente Anda, and Reidar Conradi. Effort estimation of use cases for incremental large-scale software development. In *Proceedings of the 27th International Conference on Software engineering*, pages 303–311, St. Louis, MO, USA, 2005. ACM.

Moløkken, Kjetil and Magne Jørgensen. A Review of Surveys on Software Effort Estimation. In *Proceedings of the 2003 International Symposium on Empirical Software Engineering*, page 223. IEEE Computer Society, 2003.

Moløkken, Kjetil and Magne Jørgensen. A Review of Surveys on Software Effort Estimation. *Journal of Systems and Software*, 70(1-2):37–60, 2004.

Monkevich, O. SDL-based specification and testing strategy for communication network protocols. In *Proceedings of the 9th SDL Forum*, Montreal, Canada, 1999.

Monteiro, Luis Felipe Salin and Kathia Marçal de Oliveira. Defining a catalog of indicators to support process performance analysis. *Journal of Software Maintenance and Evolution: Research and Practice*, 23(6):395–422, 2011.

Monteiro, Paula, Ricardo Machado, Rick Kazman, and Cristina Henriques. Dependency Analysis between CMMI Process Areas Product-Focused Software Process Improvement. volume 6156 of *Lecture Notes in Computer Science*, pages 263–275. Springer Berlin / Heidelberg, 2010.

Moore, Bob and Will Hayes. Building a Credible SCAMPI Appraisal Representative Sample. In *5th Annual CMMI Technology Conference & User Group*, Denver, Colorado, 2005. National Defense Industrial Association, Defense Technical Information Center.

Moore, Bob and Will Hayes. Practical Advice on Picking the Right Projects for an Appraisal. In *6th Annual CMMI Technology Conference & User Group*, Denver, Colorado, 2006. National Defense Industrial Association, Defense Technical Information Center.

Mordkoff, J. Toby. The Assumption(s) of Normality, 2010. URL http://www2.psychology.uiowa.edu/faculty/mordkoff/GradStats/part%201/I.07%20normal.pdf. Last accessed: 27-09-2016.

Morgenshtern, Ofer, Tzvi Raz, and Dov Dvir. Factors affecting duration and effort estimation errors in software development projects. *Inf. Softw. Technol.*, 49(8):827–837, 2007.

Moses, John and Malcolm Farrow. A Procedure for Assessing the Influence of Problem Domain on Effort Estimation Consistency. *Software Quality Control*, 11(4):283–300, 2003.

Neumayr, Bernd, Katharina Grün, and Michael Schrefl. Multi-level Domain Modeling with M-objects and M-relationships. In *Proceedings of the Sixth Asia-Pacific Conference on Conceptual Modeling - Volume 96*, pages 107–116, Wellington, New Zealand, 2009. Australian Computer Society, Inc.

Niazi, Mahmood, David Wilson, and Didar Zowghi. A maturity model for the implementation of software process improvement: an empirical study. *J. Syst. Softw.*, 74(2):155–172, 2005.

Nichols, William R., 2012. Personal Communication.

Nichols, William R., Mark Kasunic, and Timothy A. Chick. TSP Performance and Capability Evaluation (PACE): Customer Guide. Technical report, 2013.

Nidumola, S. The effect of coordination and uncertainty on software project performance: residual performance risk as an intervening variable. *Information System Research*, 6(3):191–216, 1995.

Niessink, F. and H. Vliet. Measurement Program Success Factors Revisited. *Information and Software Technology*, 43(10):617–628, 2001.

Nonaka, Makoto, Liming Zhu, Muhammad Ali Babar, and Mark Staples. Project delay variability simulation in software product line development. In *Proceedings of the 2007 International Conference on Software Process*, pages 283–294, Minneapolis, MN, USA, 2007. Springer-Verlag.

Oliveira, Adriano L. I., Petronio L. Braga, Ricardo M. F. Lima, and Márcio L. Cornélio. GA-based method for feature selection and parameters optimization for machine learning regression applied to software effort estimation. *Information and Software Technology*, 52(11):1155–1166, 2010.

OMG, . Software & Systems Process Engineering Meta-Model Specification 2.0, 2008. URL http://www.omg.org/spec/SPEM/2.0/. Last accessed: 27-11-2013.

Over, James. Introduction to the Team Software Process. In *SEPG Europe*, Porto, 2010. CMU/SEI.

Palza, E., C. Fuhrman, and A. Abran. Establishing a generic and multidimensional measurement repository in CMMI context. In *Software Engineering Workshop, 2003. Proceedings. 28th Annual NASA Goddard*, pages 12–20, 2003.

Park, Robert E., Wolfhart B. Goethert, and William A. Florac. Goal-Driven Software Measurement - A Guidebook. Technical report, CMU/SEI, August 1996 1996.

Patil, Mala V. Software effort estimation and risk analysis - a case study. In *Information and Communication Technology in Electrical Sciences (ICTES 2007), 2007. ICTES. IET-UK International Conference on*, pages 1002–1007, 2007.

Pestana, Maria Helena and João Nunes Gageiro. *Análise de Dados para Ciências Sociais - A complementaridade do SPSS*. Edições Sílabo, 5^{a} edition, 2008.

Pfleeger, Shari Lawrence, Ross Jeffery, Bill Curtis, and Barbara Kitchenham. Status Report on Software Measurement. *IEEE Softw.*, 14(2):33–43, 1997.

Phillips, Mike, 2010a. Personal Communication.

Phillips, Mike. CMMI V1.3 Planned Improvements. In *SEPG Europe 2010*, Porto, Portugal, 2010b. CMU/SEI.

Porter, Adam A., Jr. Votta, Lawrence G., and Victor R. Basili. Comparing Detection Methods for Software Requirements Inspections: A Replicated Experiment. *IEEE Transactions on Software Engineering*, 21(6):563–575, 1995.

Pricope, Simona and Lichter Horst. Towards a Systematic Metric Based Approach to Evaluate SCAMPI Appraisals. In *Product-Focused Software Process Improvement 10th International Conference, PROFES 2009*, volume 32, pages 261–274, Oulu, Finland, 2009. Springer Berlin Heidelberg.

Pyster, Arthur. What Beyond CMMI Is Needed to Help Assure Program and Project Success? Unifying the Software Process Spectrum. volume 3840 of *Lecture Notes in Computer Science*, pages 75–82. Springer Berlin / Heidelberg, 2006.

Rackzinski, B. and B. Curtis. Software Data Violate SPC's Underlying Assumptions. *IEEE Software*, 25(3):49–50, 2008.

Radice, Ron. Statistical Process Control in Level 4 and Level 5 Software Organizations Worldwide. In *Software Technology Conference*. CMU/SEI, 2000.

Radice, Ron. SCAMPI^{SM} with SW-CMM®. In *15th Software Engineering Process Group Conference*, Boston, Massachussets, 2003. CMU/SEI.

Ragland, Bryce. Measure, Metric, or Indicator: What's the Difference? *CROSSTALK The Journal of Defense Software Engineering*, 1995.

Raz, T. and S. Globerson. Effective sizing and content definition of work packages. *Project Management Journal*, 29(4):17–23, 1998.

Rodriguez, Daniel. Open Research Datasets in Software Engineering, 2012. Last accessed: 05-11-2011.

Rogers, Eric M. The Aims of Science Teaching. In *Nuffield Physics I*, London, 1966. Longmans.

Sarantakos, S. *Social Research*. Macmillan, South Melbourne, 1993.

Sargut, K. U. and O. Demirörs. Utilization of statistical process control (SPC) in emergent software organizations: pitfalls and suggestions. *Software Quality Journal*, 14(5):135–157, 2006.

Sasao, Shigeru, William Nichols, and James McCurley. Using TSP Data to Evaluate Your Project Performance. Technical report, 2010.

Sassenburg, Hans. Standard Investigation Method for Benchmarking IT Organisations (SIMBIO), 2009.

Sassenburg, Hans and L. Voinea. Does Process Improvement Really Pay Off? In *SEPG Europe*, Porto, Portugal, 2010. CMU/SEI.

Schaeffer, Mark. DoD Systems Engineering and CMMI. In *CMMI Technology Conference and User Group*, 2004.

Schneider, G. Michael, Johnny Martin, and W. T. Tsai. An experimental study of fault detection in user requirements documents. *ACM Trans. Softw. Eng. Methodol.*, 1(2):188–204, 1992.

Schreb, David. Accelerated Improvement Method (AIM). Technical report, CMU/SEI, May 2010 2010.

Schwaber, Ken and Jeff Sutherland. *The Scrum Guide*. Scrum.org, 2016.

SEI, CMU. SEMA, 2016. URL http://www.sei.cmu.edu/about/organization/softwaresolutions/sema.cfm. Last accessed: 11-01-2016.

SEI, CMU and SSCI. Insights on Program Success. Technical Report CMU/SEI-2009-SR-023, CMU/SEI, October 2009 2009.

Seo, Yeong-Seok, Kyung-A Yoon, and Doo-Hwan Bae. An empirical analysis of software effort estimation with outlier elimination. In *Proceedings of the 4th international workshop on Predictor models in software engineering*, pages 25–32, Leipzig, Germany, 2008. ACM.

Shirai, Yasutaka, William Nichols, and Mark Kasunic. Initial evaluation of data quality in a TSP software engineering project data repository. In *Proceedings of the 2014 International Conference on Software and System Process*, pages 25–29, Nanjing, China, 2014. ACM.

Smith, R. K., J. E. Hale, and A. S. Parrish. An empirical study using task assignment patterns to improve the accuracy of software effort estimation. *Software Engineering, IEEE Transactions on*, 27(3):264–271, 2001.

Spirula Member, , 2010. Personal Communication.

Srinivasan, K. and D. Fisher. Machine learning approaches to estimating software development effort. *Software Engineering, IEEE Transactions on*, 21(2):126–137, 1995.

Stoddard, Robert, Kevin Schaaff, Rusty Young, and David Zubrow. SEI Webinar: A Mini-Tutorial for Building CMMI Process Performance Models, 2009.

Sunetnanta, Thanwadee, Ni-On Nobprapai, and Olly Gotel. Quantitative CMMI Assessment for Offshoring through the Analysis of Project Management Repositories . In *Software Engineering Approaches for Offshore and Outsourced Development Third International Conference, SEAFOOD 2009*, volume 35, pages 32–44, Zurich, Switzerland, 2009. Springer Berlin Heidelberg.

Tabachnick, Barbara G. and Linda S. Fidel. *Multivariate Statistics*. Allyn & Bacon, Inc., Needham Heights, MA, 4 edition, 2000.

Takara, Adriano, Aletéia Xavier Bettin, and Carlos Miguel Tobar Toledo. Problems and Pitfalls in a CMMI level 3 to level 4 Migration Process. In *Sixth International Conference on the Quality of Information and Communications Technology*, pages 91–99, 2007.

Tamura, Shurei. Integrating CMMI and TSP/PSP: Using TSP Data to Create Process Performance Models. Technical Report CMU/SEI-2009-TN-033, CMU/SEI, November 2009 2009.

Tarhan, A. and O. Demirors. Investigating Suitability of Software Process and Metrics for Statistical Process Control. In *Lecture Notes in Computer Science. EuroSPI'06-Joensuu, Finland. Proceedings of 2006 European Conference on*, volume 4257, pages 88–99, 2006.

Ven, A. H.Van de and D. L. Ferry. *Measuring and Assessing Organizations*. Wiley, New York, 2006.

van Koten, Jim and A. R. Gray. An application of Bayesian network for predicting object-oriented software maintainability, 2005. Last Accessed: 31-07-2016.

van Vliet, Hans. *Software Engineering: Principles and Practice*. Willey, New York, 2007.

Venkatachalam, A. R. Software cost estimation using artificial neural networks. In *Neural Networks, 1993. IJCNN '93-Nagoya. Proceedings of 1993 International Joint Conference on*, volume 1, pages 987–990, 1993.

Vosburgh, J., B. Curtis, R. Wolverton, B. Albert, H. Malec, S. Hoben, and Y. Liu. Productivity factors and programming environments. In *Proceedings of the 7th International Conference on Software Engineering*, pages 143–152, Orlando, Florida, United States, 1984. IEEE Press.

Walia, Gursimran S. and Jeffrey C. Carver. Development of Requirement Error Taxonomy as a Quality Improvement Approach: A Systematic Literature Review. Technical Report MSU-070404, Department of Computer Science and Engineering, 2007.

Walston, C. E. and C. P. Felix. A method of programming measurement and estimation. *IBM Systems Journal*, 16(1):54–73, 1977.

Wang, Q. and M. Li. Measuring and Improving Software Process in China. In *Proceedings of International Symposium on Empirical Software Engineering - ISESE 2005*, pages 183–192, Hoosa Head, Australia, 2005.

Webb, David R., Dr. Gene Miluk, and Jim Van Buren. CMMI Level 5 and the Team Software Process. *CROSSTALK The Journal of Defense Software Engineering*, pages 16–21, 2007.

Wheeler, D. J. and R. S. Poling. *Building Continual Improvement: A Guide for Business*. SPC Press, 1998.

Wilson, David N., Tracy Hall, and Nathan Baddoo. A framework for evaluation and prediction of software process improvement success. *J. Syst. Softw.*, 59(2):135–142, 2001.

Wu, S. and Kuan Iok. A Component-Based Approach to Effort Estimation. In *Wireless Communications, Networking and Mobile Computing, 2008. WiCOM '08. 4th International Conference on*, pages 1–7, 2008.

Wu, S. I. K. The quality of design team factors on software effort estimation. In *Service Operations and Logistics, and Informatics, 2006. SOLI '06. IEEE International Conference on*, pages 6–11, 2006.

Yahya, M. A., R. Ahmad, and Lee Sai Peck. Effects of software process maturity on COCOMO II effort estimation from CMMI perspective. In *Research, Innovation and Vision for the Future, 2008. RIVF 2008. IEEE International Conference on*, pages 255–262, 2008.

Yeong-Seok, Seo, A. Yoon Kyung, and Bae Doo-Hwan. Improving the Accuracy of Software Effort Estimation Based on Multiple Least Square Regression Models by Estimation Error-Based Data Partitioning. In *Software Engineering Conference, 2009. APSEC '09. Asia-Pacific*, pages 3–10, 2009.

Yin, Robert K. *Case Study Research Design and Methods*. Applied Social Research Methods Series. SAGE, fourth edition, 2009.

Zubrow, David, Oksana Schubert, and Mark Kasunic. Consortium for Performance Measurement and Benchmarking. In *PSM User's Conference*, Vail, Colorado, 2006. SEI/CMU.

Zuse, Horst. *A Framework of Software Measurement*. Walter de Gruyter & Co., 1997.

Appendix A

Effort Estimation Methods

In this section we present the results of the literature review we conducted in order to identify the factors that influence EEA, effort estimation methods and metrics used to validate them. At the end of the appendix we map the numbers in the tables with the corresponding authors' reference.

A.1 Effort Estimation Methods

Effort estimation methods classification from our literature review on effort estimation (3.4 Effort Estimation):

Table A.1: Effort Estimation methods.

Name	**Classification**	**Ref.**	**Comments**
COCOMO I and II	model based [4] statistical model [7]	[2, 4, 7, 8] [9-11]	needs recalibration [2] [12], validity of some factors in our days is questionable [12] Boehm, 1984, 1988 [7]
FPA Metrics	model based [4]	[4, 10, 13]	
FPA		[8, 14]	"based on metric using user specifications, such as number of inputs, master files, number of logical files, number of interfaces and number of outputs to estimate software size.[14]"
MK II FPA		[8]	
Capacity Related and Price-to-Win	not "pure"	[4]	
Delphi		[10]	

Continued on next page

Table A.1 – *Continued from previous page*

Name	**Classification**	**Ref.**	**Comments**
Price-S		[15]	
SEER-SEM		[15]	
Putnam's SLIM model	statistical model	[7, 10, 15]	Boehm, 1984; Putnam, 1978
Putnum		[13]	
Doty model	statistical model	[7, 9-11]	Boehm, 1984; Herd, 1977
Bailey + Basili Meta model	statistical model	[7, 9, 10]	Bailey&Basili, 1981; Boehm, 1984
TRW model	statistical model	[7]	Boehm, 1984; Wolverton, 1974 [7]
Halsted Equation		[9-11]	
Walston-Felix		[9][11]	
Anish Mittal		[10]	
Swarup		[10]	
Least Squares Regression (LSR)		[2, 16, 17]	"generates regression model based on statistic minimizes the sum of squared errors to determine the best estimates for coefficients[9].[16]"
Expert Judgement		[4, 5]	"there is no evidence that formal estimation models are more accurate [4]"
Top-down and Bottom-up		[4]	"can be used in combination with other methods [4]"
Use Case Based	model based	[4]	
Use Case Points (UCP)		[8]	"The Use Case Points (UCP) estimation method introduced in 1993 by Karner estimates effort in person-hours based on use cases that mainly specify functional requirements of a system [11] [12]. Use cases are assumed to be developed from scratch, be sufficiently detailed and typically have less than 10-12 transactions." The method "is an extension of the Function Points Analysis and MK II Function Points Analysis [21]. [8]"
Artificial neural networks	machine learning	[2]	

Continued on next page

Table A.1 – *Continued from previous page*

Name	**Classification**	**Ref.**	**Comments**
Fuzzy Logic (FL)	machine learning	[5, 9]	GP, COCOMO and PSO have almost similar properties. FL has the lowest MMRE.
Neural Networks	machine learning	[5, 16]	"The neural network with hidden layers allows the non-linear mapping function between the causing input factors and output results. (Jun and Lee, 2001)"
Radial Basis Function (RBF) neural networks	machine learning	[18]	
MLP neural networks machine learning		[18]	"Multi-layer perceptron – applied in classification, regression and time series forecasting. [18]"
Wavelet neural networks	machine learning	[18]	
Genetic Programming (GP)	machine learning	[5, 9, 18]	
Genetic Algorithm	machine learning	[18]	
Regression Trees *Multiple additive regression trees*	machine learning machine learning	[5] [18]	"Model Trees – machine learning method for classification and regression. The leaves perform linear regression functions. Produce more understandable results than MLP."
Case-based Reasoning	machine learning	[5]	"attempts to seek a solution of the most similar past case(s), and modifies the solution considering the differences from the new target case (Jun and Lee, 2001)." Analogy with past projects (Morgenshtern et al., 2007)
Bagging predictors	machine learning	[18]	
Support vector regression (SVM)	machine learning	[18]	"based on statistical learning theory. Outperformns radial basis functions neural networks (RBFN) for software effort estimation in NASA projects' data. [18]"
Hierarchical Bayesian inference		[19]	
Bayesian Network		[16]	

Continued on next page

Table A.1 – *Continued from previous page*

Name	Classification	Ref.	Comments
Particle Swarm Optimisation (PSO)		[9, 11]	
		Features Selection	
Feature Subset Selection (FSS)		[15]	feature subset selection (FSS) and extrapolation, the selection and effort estimation is based on software parts
Effort Unit Matrix		[13]	Effort estimation for php forms, databases and documents
GA for feature selection		[18]	Reduces number of input features.
		Analogy-based Tools	
ESTOR		[11]	Size
ANGEL		[11]	Size
ACE		[11]	Size
COCONUT		[11]	Search a and b parameters in COCOMO I

A.2 Factors Related with the Process

The following table presents the factors that can be considered on the effort estimation process.

Table A.2: Factors considered on effort estimation.

Name	Definition	Ref.	Comments
Expert judgment skills	"ability to estimate the development effort of a software project applying judgement-based estimation methods"	[22]	Estimation ability factor

Continued on next page

Table A.2 – *Continued from previous page*

Name	**Definition**	**Ref.**	**Comments**
Flexibility in product and process execution	"If the project has a flexible scope, a simplification of the product can compensate for initially poor estimates and thus reduce estimation complexity and risk."	[22]	Estimation Complexity factor Incurring in the risk of being criticised for our decision, we will consider flexibility in product and process execution as a controllable factor, in particular flexibility in product. This is the context of agile development, products that can change as they evolve. So we consider that this is a controllable factor.
Inconsistent use of terminology	"When there is a lack of clear definitions of terms and there exist differences in interpretations of important estimation terminology, variance in estimation error cannot automatically be attributed to variance in estimation ability or estimation complexity."	[22]	We will try as much as possible to identify if the estimate includes a buffer and quantify it; is a weighted average of optimist, pessimist and most probable or simply represents 'most likely effort'.
Analysts capability: acap(COCOMO(C) I, II) Estimator experience from similar projects	Staff skill level [7] Estimator experience [6]	[6, 7, 12, 22] [26-29]	Increase this to decrease effort [12] Lower duration estimation error when considered the experience in the specific application area in number of projects [6]
Programmer capability: pcap (CI,II) Programmer qualifications	Staff skill level [7] Cumulative experience [30]	[7, 12, 26, 27] [28-31]	Increase this to decrease effort [12]

Continued on next page

Table A.2 – *Continued from previous page*

Name	**Definition**	**Ref.**	**Comments**
Application experience: aexp (CI, II) Programmer experience with application Team experience Staff	skill level [7] Cumulative experience [30] Project uncertainty [6]	[6, 7, 12, 26] [27-29] [30-32]	Increase this to decrease effort [12]
Modern programming practices: modp (CI)	Project requirement [7]	[7, 12, 26, 28] [29, 32-34]	Increase this to decrease effort [12] Productivity increases with "with the high use of top-down design, modular design, design reviews, code inspections, and quality-assurance programs [33]" Increases productivity [34]
Use of software tools: tool (CI, II)	Project requirement [7]	[7, 12, 26, 27] [28, 29, 31, 34]	Increase this to decrease effort [12] Increases productivity [34]
Virtual machine experience: vexp (CI)	Staff skill level [7]	[7, 12, 26, 32] [28, 29]	Increase this to decrease effort [12]
Language experience: lexp (CI) Language and tool experience (CII) Programming language experience Programmer experience with language	Staff skill level [7] Cumulative experience [30]	[7, 12, 26, 27] [28-30] [31, 32]	Increase this to decrease effort [12]

Continued on next page

Table A.2 – *Continued from previous page*

Name	Definition	Ref.	Comments
Schedule constraint: sced (CI) Required development schedule (CII)	Timing Project requirement [7] Resource Constraints [33]	[7, 12, 27, 28] [29, 31, 33, 35]	
Main memory constraint: stor (CI, II) Memory utilisation Main storage constraint	Computing platform [7] Resource Constraints [33]	[7, 12, 26, 28] [29, 31, 32] [33, 34]	Decrease this to decrease effort [12]
Database size: data (CI, II) Database complexity	Characteristics of products [7] Cumulative complexity [30]	[12, 26, 27] [28-30] [31, 36]	Decrease this to decrease effort [12]
Time constraint for CPU: time (CI) Execution time constraints (CII)	CPU occupancy Computing platform [7] Resource Constraints [33]	[7, 12, 26, 27] [28, 29, 31] [32, 33]	Decrease this to decrease effort [12]
Turnaround time: turn (CI) Computer turnaround time	Computing platform [7]	[7, 12, 26, 27] [28]	Decrease this to decrease effort [12]

Continued on next page

Table A.2 – *Continued from previous page*

Name	**Definition**	**Ref.**	**Comments**
Machine volatility: virt (CI) Virtual machine volatility	Computing platform [7]	[12, 26, 28, 29]	Decrease this to decrease effort [12]
Process complexity: cplx (CI) Product complexity (CII) Application process complexity Implementation complexity	Characteristics of products [7] Cumulative complexity [30] Program complexity [33] Project uncertainty [6]	[6, 7, 12, 26] [27-30] [31-33, 35]	Decrease this to decrease effort [12] Productivity decreases with higher percentage of complex code. Product related (non-controllable by project management) [33].
Required software reliability: rely (CI, II)	Characteristics of products [7]	[12, 26-28] [29, 31]	Decrease this to decrease effort [12]
Development for reusability: ruse (CII)		[31]	
Platform volatility: pvol (CII)		[31]	
Platform experience: plex (CII)		[31]	
Personnel continuity: pcon (CII)	Staff skill level [7]	[7, 29, 31, 32]	
Multisite development: site (CII)		[31]	
Documentation needs: docu (CII)		[31]	
Reuse	Project requirement [7]	[7, 26, 28, 29] [32]	
Type of project	Project requirement [7]	[7, 28]	We included this factor in the data dictionary

Continued on next page

Table A.2 – *Continued from previous page*

Name	**Definition**	**Ref.**	**Comments**
Programming language used	Project requirement [7]	[7, 27, 28, 35]	We included this factor in the data classification table
Software development mode	Project requirement [7]	[7, 26, 28]	
Number of source codes	Project requirement [7]	[26, 28, 29, 32] [35]	
Project size (functions or modules)	Project requirement [7]	[7, 35, 36]	
Use of chief programmer team	Project requirement [7] Total methodology [33]	[32][33]	
Team size	Project requirement [7] Work assignment factor [24]	[7, 24, 27, 34] [35]	Lowers productivity [34] From previous work it should increase development effort but it did not happen in the analysed data [24].
Design volatility	Characteristics of products [7]	[26, 27, 29]	
Complexity of delivered codes	Characteristics of products [7]	[7, 26, 27, 29] [35]	
Complexity of application processing	Characteristics of products [7]	[7, 27, 29, 36]	
Complexity of program flow	Characteristics of products [7] Cumulative complexity [30]	[7, 27, 29, 30] [36]	
Processing type	Characteristics of products [7]	[7, 27, 29, 36]	

Continued on next page

Table A.2 – *Continued from previous page*

Name	**Definition**	**Ref.**	**Comments**
Used algorithm	Characteristics of products [7]	[7, 27, 29, 36]	
Number of pages of documents	Characteristics of products [7]		
Number of displays and queries	Characteristics of products [7]	[32, 36]	
Number of personnel	Staff skill level [7]	[7, 26]	
Experience in similar project	Staff skill level [7]	[7, 32]	
Train and education staff Formal training	Staff skill level [7] Total methodology [33]	[7, 26][33]	
Type of computer used	Computing platform [7]	[26, 28, 32]	
Network type	Computing platform [7]	[7, 27]	
Requirement volatility Amount of requirements rewritten	User attributes [7] Requirements [33]	[27, 32, 33, 35] [37]	Productivity increases with accurate and stable requirements specification. Project related factors (under project management control) [33]
Interface complexity	User attributes [7]	[27, 32, 36]	
User participation in specification Client vs ITT specification	User attributes [7] Requirements specification [33]	[27, 32, 33]	Productivity increases with accurate and stable requirements. Project related (controllable) [33].
User originated design changes Customer initiated design changes	User attributes [7] Cumulative complexity [30]	[27, 30, 32]	

Continued on next page

Table A.2 – *Continued from previous page*

Name	**Definition**	**Ref.**	**Comments**
User experience in application Client Experience	User attributes [7] Client Interface [33]	[27, 33]	Productivity increases with experience. Product related (non-controllable) [33].
Management commitment	User attributes [7]	[27]	
Customer interface complexity	Cumulative complexity [30]	[30]	
Internal communication complexity	Cumulative complexity [30]	[30]	
External communication complexity	Cumulative complexity [30]	[30]	
Programmer experience with machine	Cumulative experience [30]	[30]	
Team previously worked together	Cumulative experience [30]	[30]	
Tree charts	Total methodology [30]	[30]	
Top down design hline Design formalism	Total methodology [30] Total methodology [30]	[30] [30]	
Formal documentation	Total methodology [30]	[30]	
Code reading	Total methodology [30]	[30]	
Formal test plans	Total methodology [30]	[30]	
Unit development folders	Total methodology [30]	[30]	
Number of resource constraints	Resource constraints [33]	[33]	Productivity decreases with the presence of two or more resource constraints. Product related (non-controllable) [33]
Size		[33, 34]	Productivity decreases as the number of development statements increases. Product related (non-controllable) [33]

Continued on next page

Table A.2 – *Continued from previous page*

Name	**Definition**	**Ref.**	**Comments**
Client participation	Client Interface [33]	[33]	Productivity increases with participation. Product related (non-controllable) [33]
HW concurrent with SW development		[33]	Productivity decreases with concurrent hardware development. Project related factor (controllable) [33]
Development computer size		[33]	Productivity increases as computer size increases. Project related factor (controllable) [33]
Personnel experience		[33]	Productivity increases with more experienced programming personnel. Project related factor (controllable) [33]
Project duration		[34]	Lowers productivity [34]
Execution time constraints		[34]	Lower time constraints increase productivity [34]
Moving window	Historical data [38]	[38]	Considering the chronology of projects when using historical data [38]
Level of detail		[6]	Planning at a more detailed level (shorter activities, smaller tasks) results in better data for estimation and reduces statistical errors [6]
Defects		[37]	To estimate rework
Rework		[37]	
Unclear project definition	Project uncertainty [6]	[6]	
Low project importance	Project uncertainty [6]	[6]	
Technology uncertainty	Project uncertainty [6]	[6]	
Estimation goals	Estimation development [6]	[6]	
Team focused processes	Estimation development [6]	[6]	
Participation of other groups	Estimation development [6]	[6]	

Continued on next page

Table A.2 – *Continued from previous page*

Name	**Definition**	**Ref.**	**Comments**
Concurrency	Work assignment factor [24]	[24]	"the degree to which those team members work together or separately" "increased concurrency (reflecting a higher degree of team collaboration) resulted in greater development effort". However, working together increases effectiveness. Allowing team members to focus on a smaller number of tasks improves effort. [24]
Intensity	Work assignment factor [24]	[24]	Degree of schedule compression, i.e. "to which a module's development schedule is expedited. A module with a high intensity level was worked on with sharp focus and few or no hiatuses, while a low intensity level would be associated with a module that may have sat untouched for long periods of time." More compression of the development schedule of modules improves effort. "development effort was found to decrease as intensity increased." When intensity is too high then it may have the opposite effect [24]
Fragmentation	Work assignment factor [24]	[24]	"degree to which team members' time is fragmented over multiple modules" "development effort is found to increase with fragmentation." Breaking down work to tasks that can be accomplished individually improves development effort. [24]
Actors classification	UPC estimation factor [8]	[8]	
Use cases classification	UPC estimation factor [8]	[8]	Based on average of transactions[24]
Number of new or modified actors	UPC additional factors [8]	[8]	
Transaction Alternative flow	UPC additional factors [8] UPC additional factors [8]	[8] [8]	Each counted as one use case Each counted as one use case

Continued on next page

Table A.2 – *Continued from previous page*

Name	**Definition**	**Ref.**	**Comments**
Special rules for exceptional flows, parameters and events	UPC additional factors [8]	[8]	
Number of points for modification use cases	UPC additional factors [8]	[8]	
		Scale Factors (five)	
Predecentedness : prec (CII)		[31]	Previous experience of the organisation
Development flexibility: flex (CII)		[31]	Degree of flexibility in the development process
Risk resolution: resl (CII)		[31]	Extent of risk analysis carried out
Team cohesion: team (CII)		[31]	How well they know each other and work together
Process maturity: pmat (CII)		[31]	Process maturity of the organisation

A.3 Factors Related with the Project Execution

The next table presents the factors that can cause effort estimation deviations.

Table A.3: Factors causing effort estimation deviations.

Name	**Definition**	**Ref.**	**Comments**
Accuracy of an estimation model		[22]	Estimation ability factor
Project management (cost control) ability	to manage the project to the budget [22]	[22]	Estimation complexity factor
Project member skill		[22]	Estimation complexity factor

Continued on next page

Table A.3 – *Continued from previous page*

Name	**Definition**	**Ref.**	**Comments**
Completeness and certainty of information	"measurement error of input variable [22]"	[22]	Estimation complexity factor
Inherent project execution complexity	"Innovative projects, e.g., utilizing "leading edge" technology, and projects developing complex functionality, are inherently more difficult to estimate than repeating or simple projects. Another example of inherent project complexity is size (large projects are more difficult to estimate). [22]"	[22]	Estimation complexity factor
Project priorities	"Projects with a strong focus on time-to-market, for example, typically have less accurate estimates than those with a focus on cost control. [22]"	[22]	Estimation complexity factor

Continued on next page

Table A.3 – *Continued from previous page*

Name	**Definition**	**Ref.**	**Comments**
Logging problems	"Lack of proper logging routines for the actual use of effort may result in there being differences in activities included in the measured actual effort, or may affect whether overtime is recorded or not. [22]"	[22]	Measurement process factor We will have to exclude from our analysis projects were those problems occur. This is a threat to the execution of the case study itself, the organisation may not have sufficient projects were time is accurately logged and choosing only the only the ones that do it can bias the study, because those may be the single projects that use certain effort estimation methods that require more discipline.
Difference between planned and actual output/process	"Software projects may experience increases or reductions in functionality. Similarly, the project may not conduct all planned quality assurance activities or deliver the planned quality. Differences in estimation error may be caused by these differences between planned and actual output/process and not, for example, estimation ability. [22]"	[22]	Measurement process factor We will verify differences between planned functionalities and actual functionalities (compare proposal, requirements and delivery, ask confirmation to project members), quality activities (verify verification activities and ask confirmation to testers).
Design tool		[39]	Good design tools increase productivity (generation of code).
New project members in the middle of project		[39]	Generally slows down projects due to the learning curve

Continued on next page

Table A.3 – *Continued from previous page*

Name	**Definition**	**Ref.**	**Comments**
Availability of resources	Project uncertainty [6]	[6]	
Instability Project uncertainty [6]		[6]	
Client preparedness	Project uncertainty [6]	[6]	
Customer control	Estimation management [6]	[6]	
IT unit control	Estimation management [6]	[6]	
Reporting frequency	Estimation management [6]	[6]	
Team performance assessment	Estimation management [6]	[6]	
Risk assessment	Estimation management [6]	[6]	
Functionalities		[22]	Planned and actually delivered
Defects	Source of unplanned work[37]	[37]	
Realistic expectations	Client	[40]	
Frequency of plan update		[40]	
Frequency of progress control		[40]	

The list of authors referenced to in the previous tables is the following:

[1] Jørgensen and Shepperd (2007)

[2] MacDonell and Shepperd (2007)

[4] Moløkken and Jørgensen (2003)

[5] Lopez-Martin (2011)

[6] Morgenshtern et al. (2007)

[7] Jun and Lee (2001)

[8] Mohagheghi et al. (2005)

[9] Alaa and Al-Afeef (2010)

[10] Kumar et al. (2011)

[11] Hari and Prasad Reddy (2011)

[12] Menzies et al. (2005a)

[13] Patil (2007)

[14] Wu (2006)

[15] Menzies et al. (2005b)

[16] Seo et al. (2008)

[17] Yeong-Seok et al. (2009)

[18] Oliveira et al. (2010)

[19] Moses and Farrow (2003)

[20] Jørgensen (2007)

[21] Braga et al. (2008)

[22] Grimstad and Jorgensen (2006)

[23] Jørgensen (2004)

[24] Smith et al. (2001)

[25] Foss et al. (2003)

[26] Boehm (1981)

[27] Finnie et al. (1993)

[28] Venkatachalam (1993)

[29] Srinivasan and Fisher (1995)

[30] Bailey and Basili (1981)

[31] Yahya et al. (2008)

[32] Walston and Felix (1977)

[33] Vosburgh et al. (1984)

[34] Maxwell et al. (1999)

[35] Blackburn et al. (1996)

[36] Albrecht and Gaffney (1983)

[37] Nonaka et al. (2007)

[38] Lokan and Mendes (2009)

[39] Wu and Iok (2008)

[40] Grimstad et al. (2005)

[41] Kocaguneli and Menzies (2011)

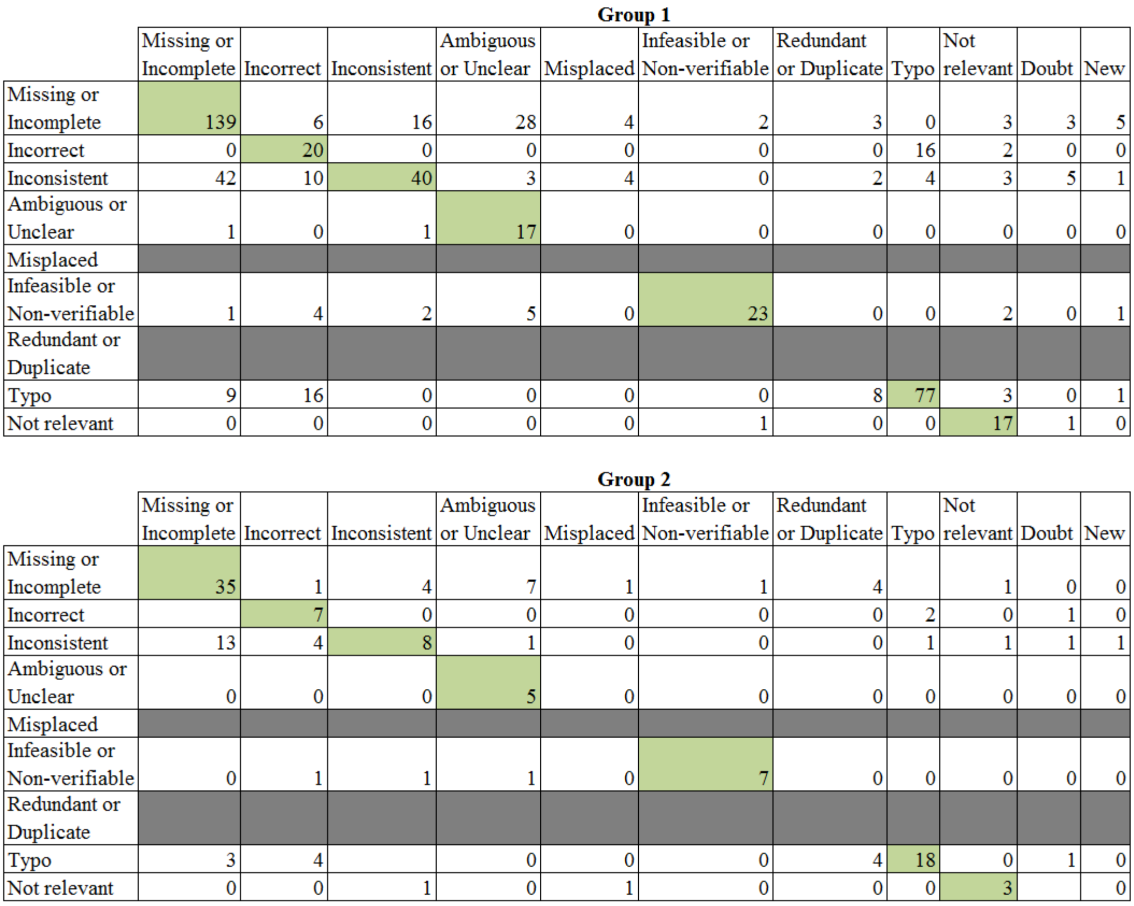
[42] Tamura (2009)

Appendix B

Process Improvement: Requirements Defects Classification

B.1 Confusion Matrix

In Figure B.1 we present the confusion matrix of the defects classifiers in the first experiment and second experiment, respectively. We can see per each one of the expected classifiers, represented in each row, which classifier was actually selected, corresponding column.



Group 1

	Missing or Incomplete	Incorrect	Inconsistent	Ambiguous or Unclear	Misplaced	Infeasible or Non-verifiable	Redundant or Duplicate	Typo	Not relevant	Doubt	New
Missing or Incomplete	139	6	16	28	4	2	3	0	3	3	5
Incorrect	0	20	0	0	0	0	0	16	2	0	0
Inconsistent	42	10	40	3	4	0	2	4	3	5	1
Ambiguous or Unclear	1	0	1	17	0	0	0	0	0	0	0
Misplaced											
Infeasible or Non-verifiable	1	4	2	5	0	23	0	0	2	0	1
Redundant or Duplicate											
Typo	9	16	0	0	0	0	8	77	3	0	1
Not relevant	0	0	0	0	0	1	0	0	17	1	0

Group 2

	Missing or Incomplete	Incorrect	Inconsistent	Ambiguous or Unclear	Misplaced	Infeasible or Non-verifiable	Redundant or Duplicate	Typo	Not relevant	Doubt	New
Missing or Incomplete	35	1	4	7	1	1	4		1	0	0
Incorrect		7	0	0	0	0	0	2	0	1	0
Inconsistent	13	4	8	1	0	0	0	1	1	1	1
Ambiguous or Unclear	0	0	0	5	0	0	0	0	0	0	0
Misplaced											
Infeasible or Non-verifiable	0	1	1	1	0	7	0	0	0	0	0
Redundant or Duplicate											
Typo	3	4		0	0	0	4	18	0	1	0
Not relevant	0	0	1	0	1	0	0	0	3		0

Figure B.1: Confusion matrix of the experiments conducted with the two groups of students.

Note: each line represents the expected classifier and the ones actually used. The diagonal is signalled in green (shadowed) to highlight the expected classifier.

Appendix C

Effort Estimation Accuracy Ranges

The boxplots of the EEAs of phases measured in LOC that show different distribution per range of total EEA are represented in Figure C.1. Those are the phases where their EEA better matches the ranges of total EEA.

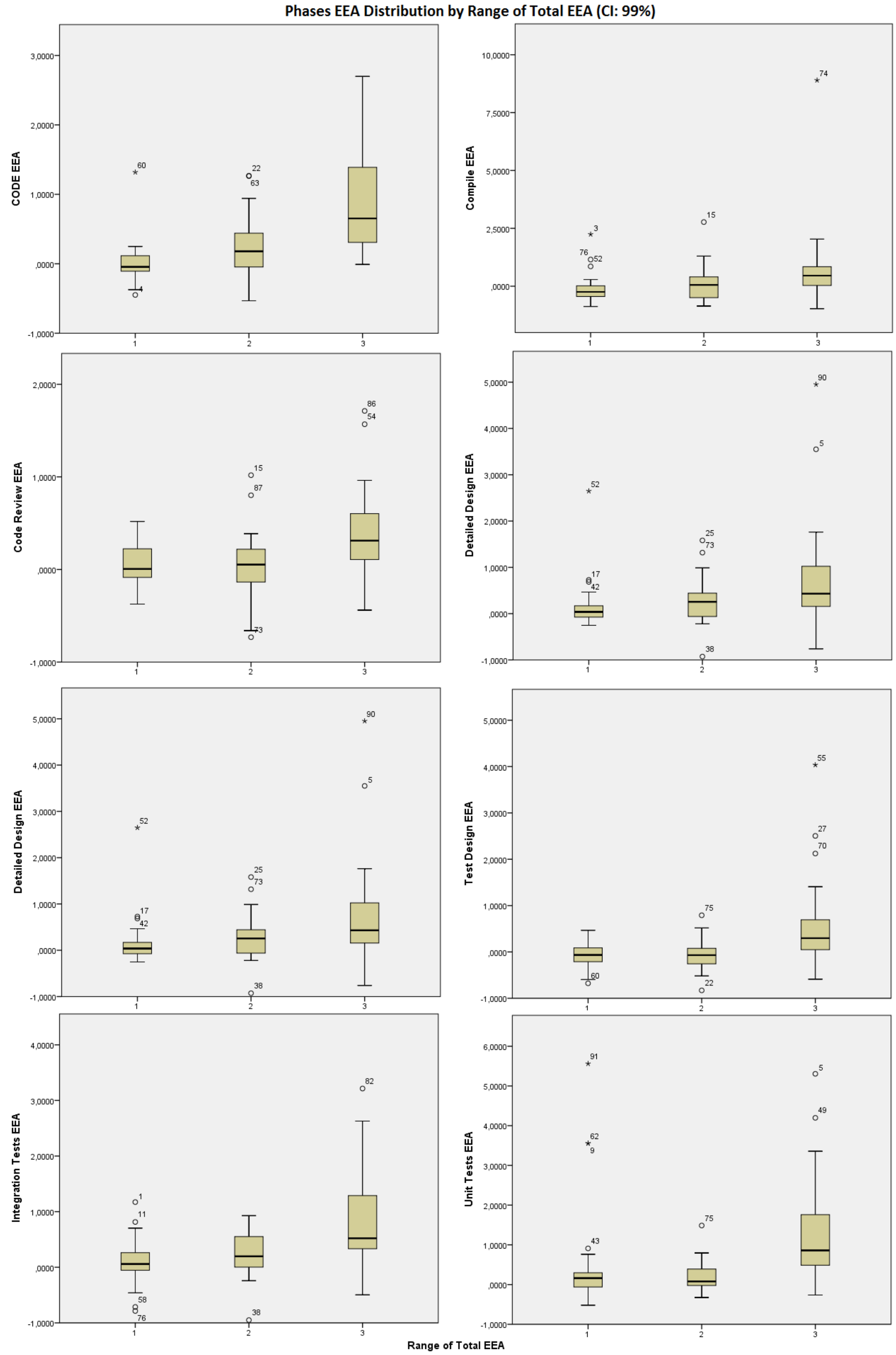


Figure C.1: Phases EEA distribution by the ranges of Total EEA: 1 - <= 10%; 2 - 10 < Total EEA <= 25%; 3 - Total EEA > 25%.

Appendix D

Survey About EQualPI

We conducted a survey regarding the usefulness of the EQualPI Framework, its requirements, and purposes. The objective of the survey was to anticipate people interest in the Framework. The survey was distributed at the SEPG Europe 2013 conference and put online in CMMI related LinkedIn groups.

D.1 Survey Questions

A Framework to Evaluate the CMMI Practices Performance

In the 2011 SEPG Europe we presented the ground theory to design a framework to evaluate the quality of implementation of the CMMI practices, which we are demonstrating. The framework will allow organisations to evaluate the performance of the practices they have in place and methods that are used to implement them. The evaluation is done using performance indicators aligned with the organisation's business goals, the leading indicators shall anticipate the course of processes and projects, while the lagging indicators appraise their performance. The purpose of our framework is to provide a self-assessment tool for organisations that want to have visibility of their processes and know if they are achieving the benefits of using CMMI.

In the case of our framework quality is defined not only by compliance (already evaluated by SCAMPI), but also by effectiveness and efficiency of the practice. In this context, effectiveness is shown by metrics that indicate that the practice benefits are met and the results of using it are the expected ones; efficiency allows organisations to make the compromise of how complex should the process be in order to achieve a goal and avoid endless and wasted effort. The evaluation results are quantitative. For example, in PP SP1.4 "Estimate effort and cost", the evaluation indicator is the deviation of the actual effort from the planned effort, expressed in percentage.

For more details please read our paper: "Towards a Framework to Evaluate and Improve the Quality of Implementation of CMMI® Practices", that can be found here:

http://paginas.fe.up.pt/ pro09003/publications.html

Our framework will allow organisations to:

- Know their performance towards reaching their business goals at any time;
- Prioritise process improvement initiatives based on the impact that certain practices or methods have on business goals;
- Predict the impact that a process improvement has in other practices performance and, ultimately, in the organisation's performance indicators and goals;
- Anticipate compliance with CMMI before SCAMPI and evaluate performance through time;
- Anticipate methods effectiveness;
- Plan to have the right levels of performance, depending on business or project demands, avoiding waste.

The following questions will allow us to receive the CMMI practitioners feedback about the framework usefulness and help us identify missing requirements. The data collected in this survey will be masked so that organisations and individuals cannot be traced back. The data will be secured and shall only be analysed by the PhD student (Isabel Margarido). The results will be presented in their aggregated form.

Thank you for your collaboration, we value your opinion!

Please feel free to contact me if you have any questions or further comments:
isabel.margarido@gmail.com

1. Demographic data:

1.1. What is your main role in your organisation?

Consultant
Faculty Member
Researcher
Software/Product/Service Developer
Project Engineer
Senior Engineer
Quality Assurance
Project Manager
Department Manager
Quality Manager
CMMI Sponsor
Upper Manager (CEO, CIO, CFO, COO, Director...)
Other (please indicate it in the comments)

1.1 Comments

1.2. What is your relation with CMMI?

Heard of it

Member of an appraisal team

Process improvement manager

CMMI Instructor

Lead appraiser

Received official CMMI training

Other (please indicate it in the comments)

1.2 Comments

1.3. What is your company name?

The name will be masked for data treatment.

1.4. What is the maturity/capability level of your organisation?

Received in a SCAMPI A. Please select 1 if an unsuccessful SCAMPI for level 2 was conducted. If you are an independent consultant or freelancer please skip this question.

Never appraised

Level 1

Level 2

Level 3

Level 4

Level 5

Waiting for appraisal (please indicate for which level in the comments box)

1.4 Comments

1.5. What is the perceived maturity/capability level of your organisation?

The one that you believe would be the current level if there were a SCAMPI A done today. If you are a freelancer or independent consultant please skip this question.

Level 1

Level 2

Level 3

Level 4

Level 5

1.5 Comments

2. About the Framework

2.1. Would you be interested in this Framework for your organisation?

Very Interested

Interested

Mildly

Maybe

Not Interested

2.1 Comments

2.2. Would your organisation be interested in this framework?

If you are an independent consultant you may skip this question.

Very Interested

Interested

Mildly

Maybe

Not Interested

2.2 Comments

2.3 How useful do you find this framework concept to evaluate organisation's results/performance?

Very Useful

Useful

Mildly Useful

Maybe

Useless

2.3 Comments

2.4 How useful do you find the framework requirements (R1 to R6)...

R1: Align quantitative business goals with performance indicators.

Very Useful

Useful

Mildly Useful

Maybe

Useless

R1 Comments

R2: Align practices with methods and performance indicators.

Very Useful

Useful

Mildly Useful

Maybe

Useless

R2 Comments

R3: Evaluate effectiveness and efficiency of practices and methods.

Very Useful

Useful

Mildly Useful

Maybe

Useless

R3 Comments

R4: Show dependencies between practices and indicators so when the methods used to implement a practice change we can anticipate its impact in the performance.

Very Useful

Useful

Mildly Useful

Maybe

Useless

R4 Comments

R5: Have indicators to anticipate projects and processes performance (leading).

Very Useful

Useful

Mildly Useful

Maybe

Useless

R5 Comments

R6: Have indicators for posterior evaluation of project or process execution (lagging).

Very Useful

Useful

Mildly Useful

Maybe

Useless

R6 Comments

2.5 How useful do you find the framework to (purposes P1 to P8)...

P1: Do a pre-SCAMPI evaluation to ensure that it will be worth doing an official appraisal.

Readiness-review

Very Useful

Useful

Mildly Useful

Maybe

Useless

P1 Comments

P2: Do process improvements.

Very Useful

Useful

Mildly Useful

Maybe

Useless

P2 Comments

P3: Select processes/practices to improve.

Very Useful

Useful

Mildly Useful

Maybe

Useless

P3 Comments

P4: Manage processes (qualitatively and quantitatively).

Very Useful

Useful

Mildly Useful

Maybe

Useless

P4 Comments

P5: Manage business goals.

Very Useful

Useful

Mildly Useful

Maybe

Useless

P5 Comments

P6: Anticipate process behaviour.

For example, deviation from the normal indicator values.

Very Useful

Useful

Mildly Useful

Maybe

Useless

P6 Comments

P7: Anticipate project behaviour.

Very Useful

Useful

Mildly Useful

Maybe

Useless

For example, deviation from the schedule.

Very Useful

Useful

Mildly Useful

Maybe

Useless

P7 Comments

P8: Forecast the achievement or not of business goals.

Very Useful

Useful

Mildly Useful

Maybe

Useless

P8 Comments

Further comments, requirements or purposes.

Please provide any further comments about the framework in general. Feel free to indicate further requirements and purposes for its usage.

Regarding the requirements R1 to R6 and Purposes P1 to P8 please indicate tools or methodologies that fulfil them.

What do you think that would be a differentiation factor in this framework?

If you wish to receive the final results of the survey please provide your e-mail.

D.2 Results

We received 25 responses to our survey. Regarding the respondents role, the majority of them were consultants and upper managements of consultancy organisations, 46% of the subjects. Their knowledge of CMMI was mainly the one of lead appraisers, 48%, and implementers, 40%. These results are depicted in the graph in Figure D.1.

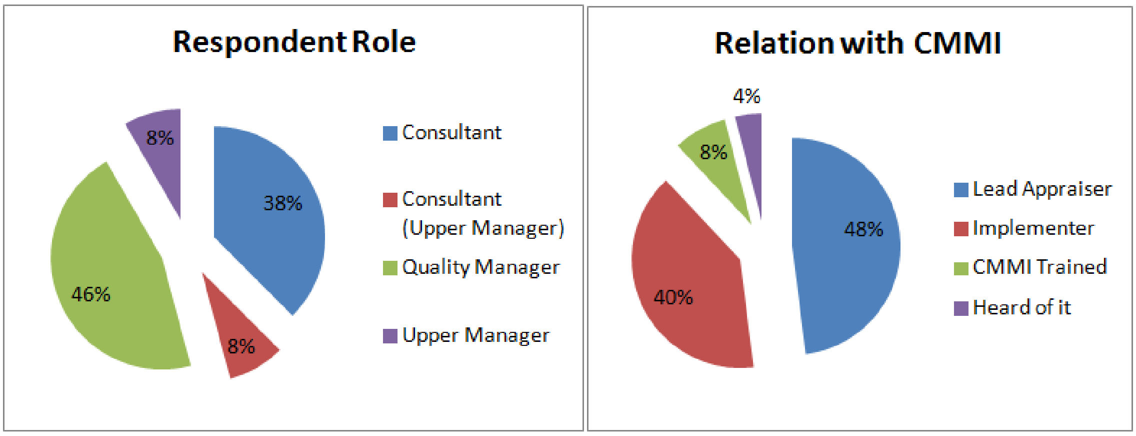


Figure D.1: Role of the subjects in the organisation and their relation with the CMMI.

In respect to the CMMI level of the organisations, the majority had never been appraised and the ones that did indicated to have a perceived level equal to the appraised one (Figure D.2).

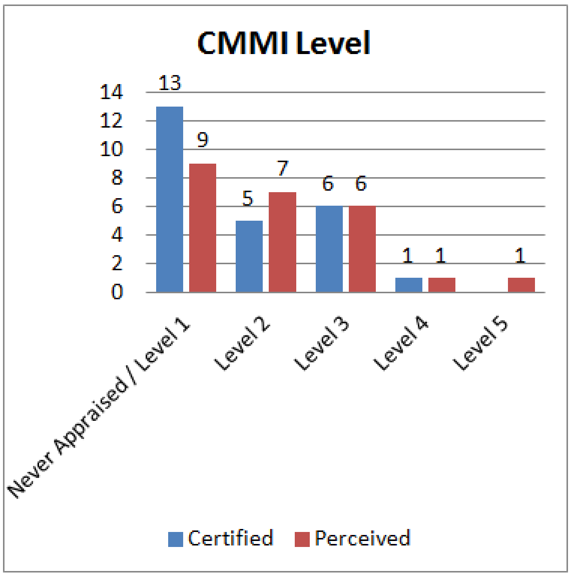


Figure D.2: CMMI level, appraised and perceived.

Regarding the interest 76% of the subjects were interested or very interested in the Framework, but only 36% considered that their organisations would be interested in it. Nonetheless, some of the consultants stated that they would be interested in the Framework to be used by their clients. 64% of the respondents considered the Framework would be useful to evaluate organisations' results and performance. The graphs to these answers are presented in Figure D.3.

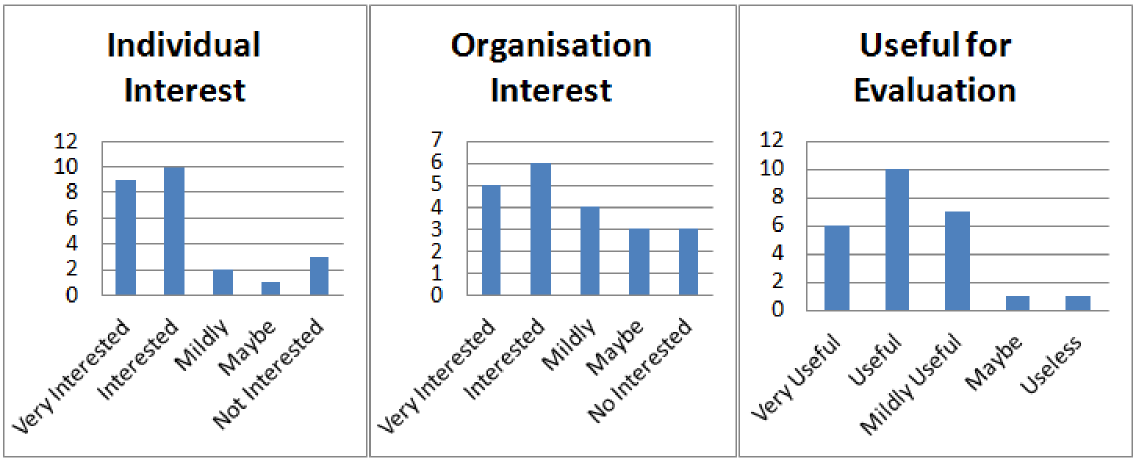


Figure D.3: Respondents and their organisations interest in the Framework and their opinion regarding its usefulness for performance evaluation.

We elicited six requirements of the Framework and asked the subjects how useful they found them, in a scale of 1 to 5, from very useful to useless. The requirements *R1. Goals and PIs alignment* and *R2. Practices and PIs alignment* were found less useful than the remainder, only 60% and 64% of the respondents found them to be useful and very useful. Both requirements *R3. Evaluate efficiency and effectiveness of the practices* and *R4. Show dependencies between them*, were found useful and very useful by 80% of the subjects. 72% of the respondents found *leading indicators* useful and very useful, while only 68% found *lagging indicators* to be useful and very useful. The graphs in Figure D.4 show their answers. Some of the respondents indicated that leading indicators and prediction were more relevant for high maturity. In our opinion the leading indicators are necessary to analyse if a practice is indeed well implemented and rendering its expected result.

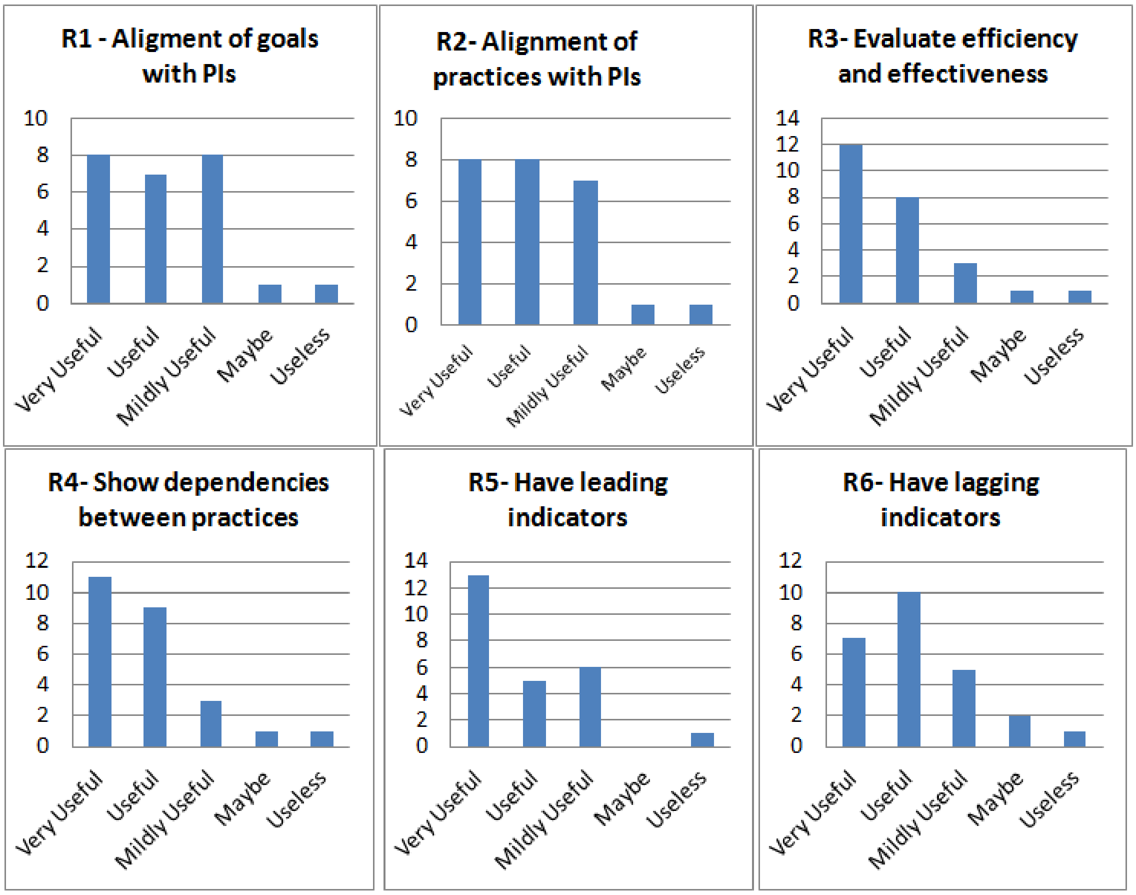


Figure D.4: Opinion of the subjects regarding the usefulness of the Framework requirements.

We also questioned the subjects about how useful they found the 8 purposes that the Framework was developed for. From doing *P1. Pre-evaluation before appraisal*, do *P2. Process improvements*, *P3. Select them based on performance* and *P4. Quantitatively manage process performance*, 72% and 76% (alternatively) of the respondents found those purposes useful and very useful. Only 60% of them found the purpose of *managing projects quantitatively* (P5) useful and very useful. Some of their comments were that the purpose was more useful for high maturity. The remainder purposes (P6 to P8) were found to be useful and very useful by 68% of the subjects. These results are represented in the graphs in Figure D.5.

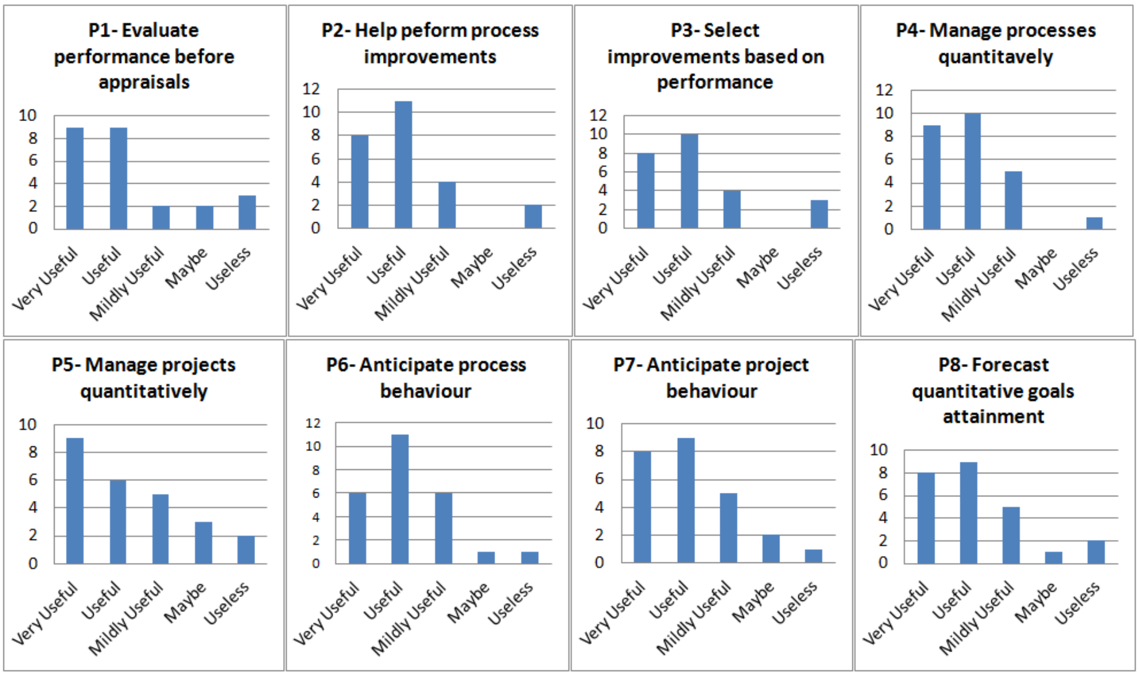


Figure D.5: Opinion of the subjects regarding the usefulness of the Framework purposes.

Some of the respondents indicated what they considered would be a differentiating factor in the framework:

"Enables to improve not just process usability/capability but even maturity/capability."
21-01-2014, Quality Manager

"I like the focus of your research and have experienced CMMI implementations where the ML3 was reached but the process performance before the customer was not improved. I think continued emphasis and measurement of effectiveness and efficiency is a differentiator for your work."
9-12-2013, Quality Manager

"Anticipate behaviour."
21-01-2014, Quality Manager

D.3 Conclusion

Some of the respondents who were consultants gave comments indicating they would rather provide their services rather than giving the organisations the tools to be able to implement CMMI on their own. Regardless, the results of the survey appear to be satisfactory has the majority of subjects found the requirements and purposes useful and very useful, and had a similar inclination regarding its usefulness to evaluate results performance. From the survey, even though the number of respondents was small we would say there is a tendency to find value in EQualPI.