Universidade de Aveiro Departamento de Economia, Gestão e Engenharia Industrial
2010

Olga Volodymyrivna Fedotova

Melhoria do Processo de Desenvolvimento de Software: uma abordagem através da inovação e gestão de conhecimento

Software Development Process Improvement: an approach involving innovation and knowledge management

Universidade de Aveiro
2010

Departamento de Economia, Gestão e Engenharia Industrial

Olga Volodymyrivna Fedotova

Melhoria do Processo de Desenvolvimento de Software: uma abordagem através da inovação e gestão de conhecimento

Software Development Process Improvement: an approach involving innovation and knowledge management

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia e Gestão Industrial, realizada sob a orientação científica da Doutora Leonor da Conceição Teixeira, Professora Auxiliar no Departamento de Economia, Gestão e Engenharia Industrial da Universidade de Aveiro e co-orientação científica da Doutora Helena Maria Pereira Pinto Dourado e Alvelos, Professora Auxiliar no Departamento de Economia, Gestão e Engenharia Industrial da Universidade de Aveiro.

o júri

presidente

Prof. Doutor Carlos Manuel dos Santos Ferreira
Professor Associado c/ Agregação da Universidade de Aveiro

Prof. Doutor José António Rodrigues Pereira de Faria
Professor Auxiliar da Universidade de Porto (Arguente)

Profª. Doutora Leonor da Conceição Teixeira
Professora Auxiliar da Universidade de Aveiro (Orientadora)

Profª. Doutora Helena Maria Pereira Pinto Dourado e Alvelos
Professora Auxiliar da Universidade de Aveiro (Co-orientadora)

agradecimentos

A execução deste trabalho foi conseguida graças ao contributo de diversas pessoas e entidades. Mesmo correndo o risco de não as referir a todas, gostava de manifestar os meus agradecimentos:

- À minha família, pelo apoio moral concedido durante a realização deste trabalho.
- Aos meus orientadores da Universidade de Aveiro, Doutora Leonor Teixeira e Doutora Helena Alvelos, pelo acompanhamento.
- À empresa *Primavera Business Software Solutions*, onde todo este projecto decorreu, com particular destaque à Drª. Fernanda Queirós, ao Dr. Henrique Narciso e ao Mestre Paulo Cruz.

palavras-chave

Desenvolvimento de Software, Inovação, Gestão de Conhecimento, Estimação de Esforço, CMMI

resumo

Este trabalho, organizado em torno de três artigos, tem como objectivo contribuir para a melhoria do processo de desenvolvimento de software através das práticas de gestão do conhecimento (GC), da inovação e de conceitos de estimação de esforço, no âmbito de uma empresa Portuguesa líder no desenvolvimento de software – *Primavera Business Software Solutions* (*BSS*). O primeiro artigo descreve uma *framework* – *'I innovate!'*, utilizada pela *Primavera BSS*, no sentido de promover a inovação. O segundo artigo analisa a prática da GC na implementação do *Capability Maturity Model Integrated* (CMMI). Finalmente, o terceiro artigo é dedicado ao estudo dos modelos de previsão do esforço de desenvolvimento de software para as equipas de teste e de desenvolvimento, com base na técnica de Regressão Linear Múltipla.
Os referidos estudos revelaram a importância da prática de GC, da inovação e da estimação de esforço na indústria de software, no sentido de obter vantagem competitiva.

keywords

Software Develpment, Innovation, Knowledge Management, Effort Estimation, CMMI

abstract

This work is organized by three papers that intend to contribute to the software development process improvement through the application of the knowledge management (KM), innovation and effort estimation concepts in the Portuguese leader software development organization – *Primavera Business Software Solutions* (*BSS*). The first paper describes the innovation framework – *'I innovate!'*, adopted by *Primavera BSS* in order to stimulate innovation promotion. The second paper analyses the practices of KM during the implementation of the *Capability Maturity Model Integrated* (CMMI) in organization. Finally, the third paper is dedicated to the study of models of the software development effort prediction for *Primavera BSS* testing and development teams based on the Multiple Linear Regression technique.
Referred studies revealed the importance of KM, innovation and effort estimation in the software industry in a search of competitive advantage.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

COP – Communities of Practices

CMMI – Capability Maturity Model Integrated

CMMI-DEV - Capability Maturity Model Integrated for Development

CS – Change Set

GP – Generic Practice

KM – Knowledge Management

ML – Maturity Level

MLR – Multiple Linear Regression

MMRE – Mean Magnitude Relative Error

PA – Process Area

PRED – Percentage Relative Error Deviation

Primavera BSS – Primavera Business Software Solutions

SD – Software Development

SDO – Software Development Organization

SECI – Socialization, Externalization, Combination, Internalization

SEE – Software Effort Estimation

SEI – Software Engineering Institute

GENERAL INTRODUCTION

1. Introduction to the Problem and Motivation for the Topic

Nowadays successful organizations are continuously searching for new ways of improving products and services in order to guarantee sustainable competitive advantage. The strategic focus on tangible resources was changed in mid-1980s, when organizations began to understand the increasing importance of knowledge in the emerging competitive environment (Grant, 1996; Newell, Swan, & Robertson, 2000; Wiing, 1997). Since then, knowledge was considered to be the productive resource for innovation, effective competition, market place distinction, economic growth and survival (Demarest, 1997; Johannessen, Olsen, & Olaisen, 1999). In the knowledge-based strategy the knowledge bearer – worker – is viewed as the relevant and interchangeable unit of analysis, since each individual has its unique knowledge (Allee, 1997) and constitutes the core intelligence competence of the organizations (Carneiro, 2000).

Knowledge Management (KM) is the coordination of the organizational people, technology, processes and organizational structure through creating, sharing and applying knowledge (Mishra, 2009).

One of the main benefits of the effective KM includes the innovation implementation within the organization (Nonaka & Takeuchi, 1995). An empirical study, carried out by Darroch (2005) revealed that firms that well develop KM practices are likely to be more innovative.

According to Nonaka & Takeuchi (1995) such KM process, as knowledge creation, stimulates innovation within the organization, while continuous innovation leads to competitive advantage (Figure 1).

Innovation is the "iterative process aimed at creation of new products, processes, knowledge or services by the use of new or even existing knowledge" (Kusiak, 2009). Hamel (1998) in (Darroch & McNaughton, 2002) and Roberts (1998) in (Darroch &

McNaughton, 2002) emphasize the importance of innovation as the key ingredient for those organizations that want to remain competitive or to pursue long-term advantage.

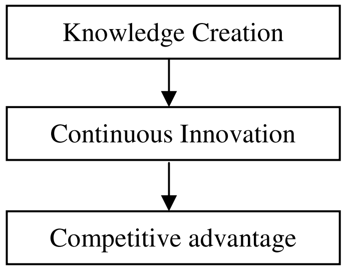

Fig.1 – Knowledge as a Competitive Resource (Nonaka & Takeuchi, 1995).

Innovation and KM are two challenging issues for the success of modern organizations that turn to be more critical in such knowledge-based industry as software development because here “people are the only true agents in business and all other assets are the results of human actions” (Sveiby, 1997). Short lifecycle of the software products, more demanding customers and global competition in this industry require its participants to look for more sophisticated ways of differentiation and competitive advantage achievement by means of innovation and knowledge management implementation oriented to the fast and original market and technological response.

As it was already mentioned, KM is the fundament of the innovation. From the other side, Wang & Ren (2006) state that the improvement of software development process leads to the implementation of knowledge management processes in the software development organizations. Thus software process improvement is essential for the knowledge management and further innovation realization within the software industry.

There are a lot of methodologies, frameworks and tools designed to improve the quality of software development process, like SCRUM, ITIL, SPICE model etc. The one used by the organization that was the object of this study is the Capability Maturity Model Integrated (CMMI) for Development.

According to Chrissis, Konrad *et al.* (2009) CMMI framework divides the company’s activity into 22 Process Areas. The Process Area (PA) is a group of related practices in the specific area that, when implemented collectively, satisfy a set of goals that are established in order to improve the process in analysis. Each process area consists of the Specific and

Generic Goals and Practices, and informative elements such as sub practices, typical work products, amplifications etc. CMMI's scope concerns the process and project management, engineering and support (Chrissis, et al., 2009; Kupla & Jonhnson, 2008).

The model provides two possible representations: the staged and the continuous (Kupla & Jonhnson, 2008). The staged model is the most well known (Staples, et al., 2007) and is currently being implemented in the analyzed organization. In order to measure the process improvement staged representation of CMMI distributes all PAs by five Maturity Levels (ML): Initial (ML 1), Managed (ML 2), Defined (ML 3), Quantitatively Managed (ML 4) and Optimizing (ML 5).

Benefits of CMMI implementation involve improvements in the quality of the end product, return on investment, schedule, productivity and customer satisfaction, among other (Gibson, Goldenson, & Kost, 2006). One of the main pillars of the Project Planning PA deals with the cost and effort estimation. Specific Practice 1.4 – *Determine Estimates of Effort and Cost* requires from organization the implementation of a formal model of cost and effort estimation in order to achieve the Specific Goal 1 – *Establish Estimates* of Project Planning PA.

Software cost and effort estimation is the "discipline that attempts to foresee the effort required for the completion of a software development project" (Sentas, Angelis, Stamelos, & Bleris, 2005). There is a great variety of techniques dedicated to estimate effort predictions, there is no single technique that would totally satisfy the needs of all software development organizations, as each of them has its weaknesses and strengths (Boehm, Abts, & Chulani, 2000).

According to Jalote (1997) "for the software development projects, detailed and accurate cost and schedule estimates are essential prerequisites for managing the project" as they are the basis of efficient decision making about the project cost, schedule and manpower allocation. Accurate estimates are among the factors that influence the success of projects. They directly impact the cost, schedule, end product quality and subsequently customer satisfaction and financial prosperity of the organization (Briand, Langley, & Wieczorek, 2000).

The focus of software development organizations on KM, innovation and definition of accurate estimations may be explained by the necessity to attend the technological progress and market changes in order to conquer and preserve favourable market positions.

The next section provides a short characterization of the software development organization that was the object of this study – Primavera Business Software Solutions (BSS).

2. Organization Characterization

Primavera BSS is a multinational medium-sized software development organization dedicated to the elaboration of business management solutions and enterprise resource planning applications. Founded in December of 1993 in Braga by Jorge Batista and Jose Dionisio, with delegations in Lisbon, Luanda and Madrid, Primavera BSS is now considered to be one of the 500 largest European companies with greatest growth potential, according to the ranking promoted by Growth Plus (Primavera-BSS, 2010).

Primavera BSS works with approximately 40.000 organizations all over the world and counts with about 150.000 users. It is leader of the Portuguese and Angolan markets, with 73% and 16% of the sales volume, respectively. Besides, organization's products are present in the markets of such countries as Brazil, Spain, Angola, Mozambique, Cape Verde, France, among other.

Primavera BSS provides a vast range of solutions for companies of different sizes (from small to large ones) and from different industries such as services, manufacturing, retail, and public administration. Among the principal offers one can find solutions in Accountancy, Human Resources, Commercial Management, Business Intelligence, Enterprise portals, Mobile Sales as well as specific solutions for the earlier defined industries. Roche, Epson, Deutsche Bank, Remax, Mazda and Chicco are some reference clients that benefit from using Primavera's products and services.

Microsoft and Business Objects are the main technological partners of Primavera BSS that provide organization with newest technological solutions and consequently contribute to its success.

Figure 2 represents the evolution of the sales volume of the organization from the 2006 until 2008. As can be seen, the sales grew 29% in the first year and 25% in the second one. From the Figure 3 can be observed that the weight of the international market on the sales volume have been growing in constant way during the time period from 2005 to 2008, being in 2008, almost one third of the total volume of sales.

These results are owing to the Primavera's trade-mark consolidation in the national market as well as to the strategic expansion of the organization to the international markets.

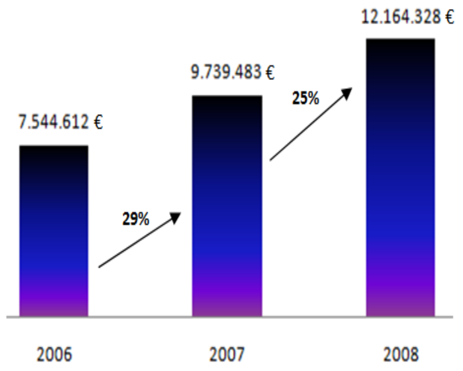

Fig.2 – Evolution of the Sales Volume.

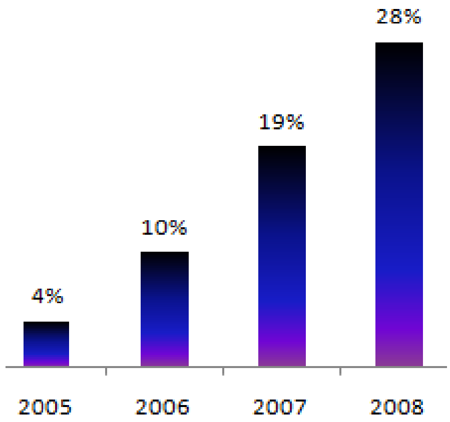

Fig.3 – Weight of International Markets.

The main values of Primavera BSS are concentrated around such concepts as innovation, sustainability, commitment, integrity, professionalism and people. The mission of the organization consists in exploring and developing new technologies that result in the innovative solutions (Primavera-BSS, 2010).

3. Objectives and Organization of the Dissertation

The principal objective of this work consists in the study of the factors that may help software development organizations to achieve competitive advantage over the rivals as well as to conquer and maintain the favourable market positions through the implementation of knowledge management, innovation and accurate software effort estimations within this type of organizations.

Innovation is one of the main requisites of the profitability of software development industry. There was carried out a study on the basis of innovation experience of Primavera

BSS, which aimed at describing the way of new ideas generation and contribution of knowledge management to this process. Chapter I of the dissertation contains the paper dedicated to the description of the framework for the innovation promotion adopted by Primavera BSS. This framework contributes to the involvement of the whole organization in the innovation process and thus allows everyone to participate in the creation of the organization's future and permits to find new business opportunities.

To guarantee sustainable innovation, software development organizations have to implement the knowledge management process as it is the basis for innovation. Chapter II presents the study which main objective is to analyse the contributions of Knowledge Management to the implementation of the best practices of CMMI in Primavera BSS.

One of the critical areas of the CMMI concerns the effort estimation. This topic is considered of the extreme relevance in software organizations because it helps to avoid the cost and effort overruns. To upgrade the process of software development from the Initial to the Managed CMMI level, organization has to adopt a formal method of effort estimation. Chapter III presents the study that describes the adoption, by Primavera BSS, of formal effort estimation models based on the Multiple Linear Regression technique for the development and testing teams, in order to perform accurate project's effort predictions necessary to deliver the software on time, within the budget and with the expected functionalities and quality.

A FRAMEWORK FOR INNOVATION PROMOTION: APPLICATION IN SOFTWARE DEVELOPMENT COMPANY

Olga Fedotova
Dep. of Economics, Management and Industrial Engineering, University of Aveiro, Aveiro, Portugal
olgafedotova@ua.pt

Paulo Cruz
Primavera BSS, Braga, Portugal
paulo.cruz@primaverabss.com

Helena Alvelos
Dep. of Economics, Management and Industrial Engineering /GOVCOPP, University of Aveiro, Aveiro, Portugal
helena.alvelos@ua.pt

Leonor Teixeira
Dep. of Economics, Management and Industrial Engineering /GOVCOPP / IEETA, University of Aveiro, Aveiro, Portugal
lteixeira@ua.pt

ABSTRACT

This paper describes the innovation framework adopted by a software development organization (SDO) in order to implement the innovation process. There are analyzed knowledge management (KM) and innovation concepts as they provide foundation for the further understanding of the framework's operation. The innovation architecture, on which the innovation process is based, enables the framework adoption by organizations from other sectors.

KEYWORDS: Innovation, innovation framework, knowledge management.

1 INTRODUCTION

Innovation is a concept that recently gained popularity among the top managers of different organizations. This recognition may be explained by the difficulty of obtaining and maintaining competitiveness in the present context. Thus, the essential question for contemporary organizations is "how to continue to be prosperous in evolving and emerging market" (Sarkar, 2007). Innovation is closely related to the KM, as both are considered to be the efficient strategic tools that provide to organizations competitive advantage and contribute to wealth creation.

This article describes the innovation framework adopted by the Portuguese leading software development (SD) organization – Primavera Business Software Solutions (BSS) – to stimulate the innovation process and implement KM concepts. This framework integrates the best practices of innovation described by Skarzynski and Gibson (2008).

In spite of being implemented in a SDO, the presented innovation framework may be easily adapted to the needs and requirements of any other industry.

2 THEORETICAL BACKGROUND

2.1 Knowledge Management Concepts

According to Aurum *et al*. (2008) knowledge is a broad and abstract notion. Knowledge has been defined in (Schneider, 2009) as the human expertise stored in a person's mind, obtained through experience and interaction with person's environment.

Nonaka (1994) distinguishes two types of knowledge: tacit/implicit and explicit knowledge. Tacit knowledge is a kind of knowledge that is stored in the people's mind in the form of memory, skills, experience, education, imagination and creativity; while the explicit knowledge exists in a form of documented processes, directives, standards or patterns and is stored in textbooks, software products and documents (Dayan & Evans, 2006; Koskinen & Vanharanta, 2002). In this context, and in accordance with Aurum *et al.* (2008), one of the objectives of KM is to transfer the implicit knowledge into explicit one, as well as to transfer explicit knowledge from individuals to groups within the organization.

Davenport and Prusak (1998) define KM as "a method that simplifies the process of sharing, distributing, creating, capturing and understanding of a company's knowledge". Dayan & Evans (2006) and Goldshtein (2004) present the KM as a discipline that aims at "maximize innovation and competitive advantage that comes from the existence of greater amount of knowledge".

The process approach of KM described by Wang and Ren (2006) involves knowledge acquisition, knowledge share, knowledge application and knowledge innovation. Demarest (1997) considers that there are four stages of KM: construction, embodiment, dissemination and use that is the final objective of the KM system. Other framework of KM process presented by Rollet (2003) includes aggregating, planning, creating, integrating, organizing, transferring, maintaining and assessing of the knowledge.

According to Ruggles and Little (1997) "knowledge management activities are adding value to organizations by enhancing innovation and innovativeness". The next section provides definition of the innovation, gives brief overview of the types of innovation as well as of the proper process of innovation, and finally presents the relationship between the KM and the innovation process.

2.2 Innovation Concepts

Innovation is a complex and multi-facet concept. There is a great variety of innovation concept definitions. Depending on the context, it may slightly change the meaning, adapting to one or another reality, but in general innovation is defined as the application of new ideas or re-arrangement of the old ones in a new and novel way (Fischer & Suarez-Villa, 1999; Sandberg, 2008; Sarkar, 2007; West & Altink, 1996).

Among the principal types of innovation, Landau and Rosenberg (1986) distinguish: innovation of new product; new process of production; the substitution of a cheaper material; an improvement in instruments or methods of doing innovation. Another classification of innovation stated in Stamm (2008) distinguishes architectural innovation, market niche innovation, regular innovation and revolutionary innovation.

According to Urabe *et al*. (1988) innovation consists of minor and major changes: extremely major changes are called radical innovation, while cumulative series of minor changes are called incremental innovation. Geroski (2005) states that incremental innovations extend and develop already existing activities, and radical innovations lead to widespread changes and disrupt with the established habits, behaviours or technologies. Davila *et al*. (2006) distinguish three main types of innovation: incremental, semi-radical and radical (Figure I.1).

Business Model

	Similar to Existing	New
New	Semi-radical	Radical
Similar to Existing	Incremental	Semi-radical

Technology

Fig.I.1 – Innovation framework (Davila, et al., 2006).

Incremental innovation is oriented to extract the maximum value from the existing products or services without a need of great changes, risks or investments. Opposite to incremental, radical innovation involves significant changes in both parameters: business model and technology. Semi-radical innovation involves substantial changes in one of the dimensions, while the other is less affected.

Kusiak (2009) defends that there is no generic process model of innovation across different organizations, but there are three main phases that should be present in each adopted/developed model in order to create value for the innovation process (Figure I.2): idea generation phase, conversion phase (consists of idea selection and development) and diffusion phase.

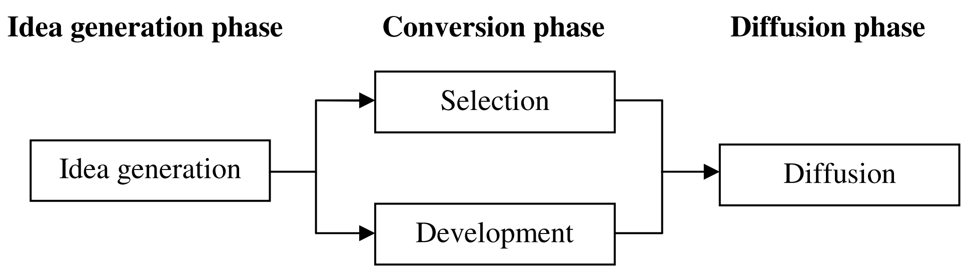

Fig.I.2 – Model of the innovation value chain (adopted from Kusiak (2009)).

Idea generation is the foundation stone of the innovation process. Each innovation starts with ideas what makes idea generation the important variable of the innovation process (Koc, 2007). Nonaka (2000) defines knowledge creation as a "continuous, self-transcending process through which one transcends the boundary of the old self into a new self by acquiring a new context, a new view of the world, and new knowledge".

The relationship between the core elements of both earlier mentioned notions (KM and innovation) is discussed in the following section.

2.3 Knowledge Creation as the Innovation engine

There are several studies that aim at defining the relationship between concepts associated to the KM and Innovation (Liao & Wu, 2009; McAdam, 2004; Popadiuk & Choo, 2006). One of them, realized by Schulze and Hoegl (2008) and based on the SECI model of knowledge creation proposed by Nonaka (1994), revealed the existing dependency between idea generation and knowledge creation.

This model identifies four modes of organizational knowledge creation (Socialization, Externalization, Combination and Internalization) that are originated from the conversion between tacit and explicit knowledge (Figure I.3).

Socialization is the tacit knowledge that results from the experience share during the informal interaction between people. Externalization corresponds to the transformation of tacit knowledge to the explicit one. The inverse process is called internalization and is known as the learning process. Combination is the re-arrangement of the existing knowledge by means of its sorting, combining, adding and categorizing (Popadiuk & Choo, 2006).

	Tacit Knowledge	**Explicit Knowledge**
Tacit Knowledge	Socialization	Externalization
Explicit Knowledge	Internalization	Combination

Fig.I.3 – Modes of knowledge creation (Nonaka, 1994).

The results of the Schulze and Hoegl (2008) study revealed that socialization and internalization are positively related to the novel idea generation, while externalization and combination have negative impact on the creation of new ideas.

2.4 Innovation and KM in a software development industry

In order to survive in the competitive environment and meet the globalization challenge, SDOs have to find their forms of operating, by adopting different frameworks for process improvement, KM and innovation. These frameworks should bring to organizations the best practices of their field that were gathered through the world organizations and re-arranged by area experts.

Globalization and technological progress make software products life cycle each time shorter. This forces industry participants to search for new, more refined solutions for the increasingly exigent customers. Thus, innovation is considered to be critical for the SDOs as it results in the strategic benefit for those who can manage it. From the other side, KM is another determinant parameter in SD industry. The success of SD is often determined by the efficient introduction and implementation of KM concepts, because the SD is made by people and is based on their expertise.

This article aims at describing the innovation framework adopted by one of the Portuguese software development organizations in order to support the innovation process and contribute to the knowledge share between the organization's collaborators, as well as to the creation of the organizational knowledge that is claimed to be its greatest asset in the struggle against the competitors.

3 CASE STUDY: INNOVATION FRAMEWORK IN A SOFTWARE DEVELOPMENT ORGANIZATION

As the software development segment is knowledge intensive, it requires dynamism and constant presentation of new solutions from its participants. In order to meet these requirements, Primavera BSS adopted the innovation framework, which formalized the process of new idea generation as well as knowledge share and in this way met one of the main objectives of KM - turn individual's knowledge into the organizational one. It is important to emphasize that before the innovation framework elaboration, Primavera's innovation process was concentrated within the one team, created on purpose to bring new technology into organization by means of its evaluation, analysis and further adoption. As the team is exclusively focused on this area, its view restricted organization only to determined kinds of innovation.

In order to follow the needs of clients and markets, a need for innovation in the other sectors of Primavera's activity was detected, thus the whole organization contribution to the innovation was satisfied by implementation of the Innovation Framework – '*I innovate*!'.

3.1 Innovation Framework – 'I innovate!'

Innovation architecture, on which the innovation framework is based, consists of four main innovation areas: product, process, marketing/business model, and organization model (Figure I.4):

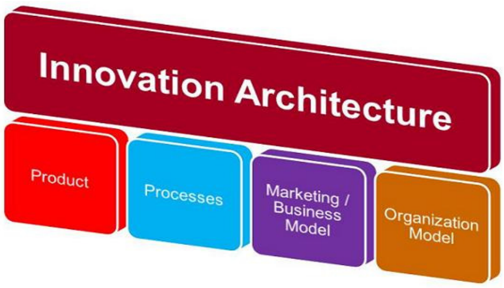

Fig.I.4 – Innovation Architecture of Primavera BSS.

All ideas, generated within the framework are organized according to this classification, allowing to stay aligned with the strategic objectives of the organization, and at the same time guaranteeing the focalization on the areas of interest. The process of innovation architecture creation is iterative and involves two phases: divergent and convergent (Figure I.5).

During the divergent phase occurs the collection of all ideas proposed by the collaborators. The more ideas organization collects during this phase, the better results it may achieve in the next phase, as the convergent phase is responsible for the ideas clustering. The clusters formed during the convergent phase provide the direction for the organization's innovation strategy and may lead to the new innovation areas creation.

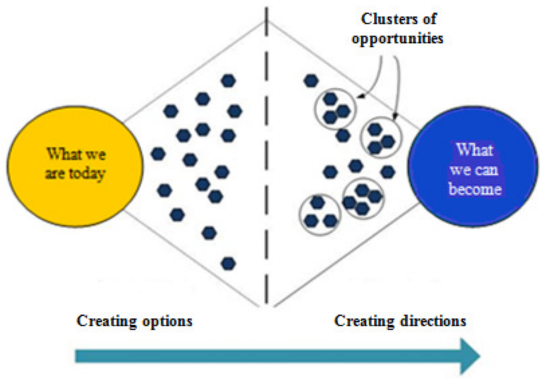

Fig.I.5 – Process of innovation architecture evolution (adopted from (Skarzynski & Gibson, 2008)).

The main objective of the innovation framework consists in adoption of the process that would allow any collaborator of Primavera BSS to provide innovation initiatives by means of 'ideation' (creation of new ideas). In this way it is possible to simplify the ideas generation within the organization and stimulate the disruptive innovation as well as to create individual and organizational knowledge.

'I innovate!' may be considered as a simple process of ideas collection, but still it has a complex mechanism behind it.

Primavera's innovation process consists of seven main activities: ideation, assessment, testing, measuring, implementation, archiving and promotion (Figure I.6).

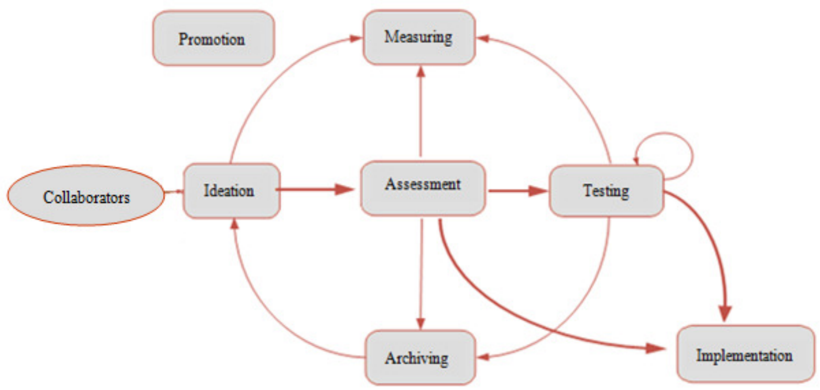

Fig.I.6 – Primavera's Innovation process.

Table I.1 provides short description of the referred activities and their equivalent to the other innovation process approaches.

Table I.1 – Primavera's Innovation activities according to different approaches.

Phases	***Primavera's Innovation Framework (Skarzynski & Gibson, 2008)***	***Innovation model (Kusiak, 2009)***	***Innovation process approach (Stamm, 2008)***
Ideation	Idea creation, registry in the innovation portal and exhibition to the collaborators for voting.	Idea generation and diffusion	Idea suggestion
Assessment	Idea evaluation according to the established criteria.	Idea selection	Idea selection
Testing	Idea validation.		
Archiving	Archiving of non-viable ideas.	-	Management of 'unsuitable' ideas
Measuring	Innovation performance evaluation.	-	-
Implementation	Idea recognition and realization.	Development	Idea Implementation
Promotion	Creation of organizational innovation culture.	-	-

Ideation

This phase consists in ideas registration by Primavera's collaborators in the innovation portal (Figure I.7).

Idea registration is the trivial task during which the collaborator exposes to the organization his/her idea. To register the idea it is necessary to provide the following data: title (obligatory); short description of the idea (obligatory); data of idea's registry (automatic); the foreseen data of idea's assessment (automatic); author's name (it is possible not to identify the author of the idea, creating an anonymous suggestion); objectives/motivation (optional); advantages (optional); and attachment (optional). After the idea is registered, it automatically becomes exposed to the community for voting during the period of one month when the author has a possibility for editing, complementing or removing it.

When the collaborator registers the idea in '*I Innovate!*' there is a conversion of individual's explicit or implicit knowledge into the organizational explicit knowledge. In this way takes place the organizational knowledge creation by means of externalization or combination described by Nonaka (1994).

Fig I.7 –Primavera's Innovation portal.

Ideation also stimulates the knowledge transformation in the opposite direction (from organizational knowledge to the individual's one). To submit the idea, any collaborator has to consult the list of already registered ideas to avoid their duplication. This process contributes to the individual knowledge creation by means of internalization and combination (Nonaka, 1994).

In this way ideation involves knowledge share between the collaborators of organization. It may also be considered as the promoter of knowledge integration because there is always an opportunity for learning from the experience and expertise of the other organization elements.

Assessment

In the end of each month, Innovation Observatory assesses and evaluates all registered ideas in terms of the potential alignment with organizational strategy, feasibility and economic value (costs, gains, revenues, etc.). The main decision factor is the alignment of the idea with the strategic objectives, which contribute in a different way to the ideas assessment. A score system is established in a way that the most relevant strategic objectives for the innovation area count as double. After the evaluation of the strategic potential of the idea, it is decided whether to proceed to the testing, implementing or archiving phase. In some cases, merging two or more similar ideas may be proposed by the Innovation Observatory to achieve a bigger scope of the resulting idea.

Testing

Testing phase usually takes place when the radical and semi-radical innovation ideas are proposed. It allows identifying and possibly mitigating the risks associated with its implementation.

To test the idea, Innovation Observatory forms a team that verifies idea's viability by means of developing a prototype (for example, in a case of new product proposal), effectuating a study or applying another type of testing. Depending on the testing results, the idea may pass to implementation phase or may be archived.

Implementation

During the implementation phase the proposed idea is turned into the organization's innovation. It may be a new product or service, improvement of the organization's processes or modifications of the business model.

Archiving

The ideas that were considered as not viable during the testing phase or not attractive from the strategic point of view are archived. These ideas may be reformulated by the authors or by any other person and newly put on voting.

Measuring

To control the process of innovation and manage the innovation value chain, it is necessary to define the metrics and indicators that will give a clear perception of the innovation progress within the organization (Kusiak, 2009). Innovation platform supports the data used for metrics elaboration.

Currently, the indicators used to measure the innovation performance of Primavera BSS deal with the percentage of ideas aligned with the strategic objectives, number of originated ideas per country, percentage of collaborators that participated in the process, medium number of ideas per month, number of projects per innovation area, framework's efficiency, etc.

The results that show the success of the innovation framework adopted by Primavera BSS in July 2009 can be observed in Figure I.8, which presents the ideas distribution according to their phase, by March 2010. For the relatively short period of framework's operating (8 months) 64 collaborators registered 107 new ideas (the established organizational objective was of 100 ideas/year). According to the data referent to March 2010, 5 of the registered ideas are already implemented, 8 are under implementation, 7 of the registered ideas were considered viable after testing, and 7 of them are under assessment.

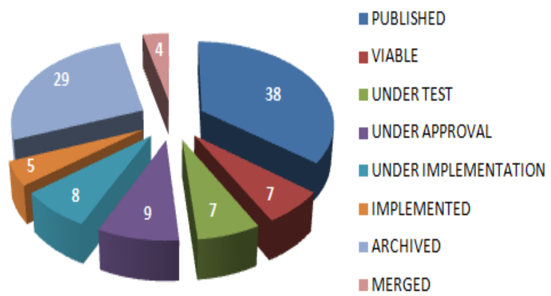

Fig.I.8 – Ideas' distribution according to the different phases (March, 2010).

Table I.2 – Ideas Distribution by type of innovation and improvement area.

Type Innovation		Improvement Process Area					
		Frequency Type	Marketing/ Business Model	Organizational Model	Process	Product	Total
	Incremental	Absolute	3	1	8	4	16
		Relative	8,33%	2,78%	22,22%	11,11%	44,44%
	Semi-radical	Absolute	6	3	5	2	16
		Relative	16,67%	8,33%	13,89%	5,56%	44,45%
	Radical	Absolute	3	0	1	0	4
		Relative	8,33%	0,00%	2,78%	0,00%	11,11%
		Total %	33,33%	11,11%	38,89%	16,67	100%

Table I.2 represents the distribution of the ideas from Figure I.8, excluding published, merged and archived ideas, according to their type (product, process, organization model, or marketing/business model) and type of innovation they are associated to (incremental, semi-radical or radical). The incremental and semi-radical ideas have the same weight of approximately 44.44% of all classified ideas. The majority of incremental ideas are related to the process improvement process area – 22.22%, while the marketing or business model improvements of Primavera BSS represent 16.67% of the semi-radical ideas. Radical ideas correspond to 11.11% of the ideas' volume and concern mainly marketing or business model of the organization – 8.33%.

Promotion

Promotion is used by the Innovation Observatory to promote the innovation framework and support the innovation process adherence by the organizational collaborators as well as encourage them to participate in the organization's future. Promotion takes place either in the form of workshops that occur monthly and are open to all elements of the Primavera group, or by other initiatives (blog, news, etc.) with the objective of incorporating a true innovation culture within the organization.

3.2 Awards and Rewards

Besides the motivations, like recognition and satisfaction of working with the preferred matter, Primavera has established to its collaborators some incentives that link rewards to performance.

One of the main incentives for the participation in the innovation process consists in the rewards and awards attribution by the innovation framework team. Once adopted, they stimulate the creativity and promote positive attitude toward changes (Tushman & O'Reilly, 2002).

Among the principal annual awards of the innovation framework are the *"Most Active Innovator"* designated for the collaborators that contributed with the major number of ideas or comments, *"Most Valuable Innovation"* and other three awards for the authors of the most successful ideas.

4 CONCLUSIONS AND RECOMMENDATIONS

Innovation and KM are key elements to the organization's success. Both of them contribute to the creation of competitive advantage. That's why and according to Mehta (2008) and Skarzynski and Gibson (2008) to make innovation and knowledge management sustainable, it is necessary to turn them into the way of life for organization, creating the correspondent environment, and motivating the creativity and initiatives of its collaborators.

The described innovation framework provides the conditions necessary for making innovation and KM happen within the organizations. In spite of being implemented in

SDO, this framework can be adapted to the needs and objectives of organizations of other type. But still, there are some improvements needed to increase the framework's efficiency as well as organization's innovation capacity.

At the moment, only the members of Primavera BSS group contribute to the innovation process of the organization. In the future it is important to involve external elements to the organization in order to contribute with new ideas that could lead to innovation's scope expanding. This aim may be achieved by opening the borders of innovation framework to the partners, institutions, users, competence centres, sector experts, among others.

Another improvement may be achieved by promoting KM and innovation framework relationship, which can be implemented by organizing the communities of practices (COP). COPs should be composed by experts of key areas and their main competencies should be (i) the discussion of the ideas registered in the innovation portal, (ii) sharing of experiences among the elements of the organization and (iii) the creation of the knowledge databases that would integrate the organization core competencies. Primavera's internal portal '*SkilLand*' may contribute to the identification of the collaborators with the similar skills and further organization of COPs as it contains the information about the organization's personnel competencies.

The archiving phase may also be improved by providing the system of revision and re-evaluation of the non-implemented ideas and their further reviving. If today the idea is out of the organization's strategic scope or is considered as non-viable it doesn't mean that tomorrow this idea will be classified in the same manner.

5 REFERENCES

Aurum, A., Daneshgar, F., & Ward, J. (2008). Investigating Knowledge Management practices in software development organisations – An Australian experience *Information and Software Technology, 50*(6), 511-533.

Davenport, T. H., & Prusak, L. (Eds.). (1998). *Working knowledge: how organizations manage what they know*. Boston, MA, USA: Harvard Business School Press.

Davila, T., Epstein, M. J., & Shelton, R. (Eds.). (2006). *Marketing innovation Work: How to Manage It, Measure It, and Profit From It*: Pearson Education, Inc.

Dayan, R., & Evans, S. (2006). KM your way to CMMI. *Journal of Knowledge Management, 10*, 69-80.

Demarest, M. (1997). Understaning Knowledge Management. *Long Range Planning, 30*, 374-384.

Fischer, M. M., & Suarez-Villa, L. (Eds.). (1999). *Innovation, networks and localities*: Springer.

Geroski, P. A. (Ed.). (2005). *Fast Second: How Smart Companies Bypass Radical Innovation to Enter and Dominate New Markets*. San Francisco: Jossey-Bass.

Goldshtein, G. Y. (Ed.). (2004). *Strategic Innovative Management*. Taganrog - Russia: Taganrog State University of Radio Engineering.

Koc, T. (2007). Organizational determinants of innovation capacity in software companies. *Computers & Industrial Engineering, 53*, 373-385.

Koskinen, K. U., & Vanharanta, H. (2002). The role of tacit knowledge in innovation processes of small technology companies. *International Journal of Production Economics, 80*(1), 57-64.

Kusiak, A. (2009). Innovation: A data-driven approach *International Journal of Production Economics, 122*, 440-448.

Landau, R., & Rosenberg, N. (Eds.). (1986). *The Positive Sum Strategy: Harnessing Technology for Economic Growth*. Washington National Academy Press.

Liao, S. H., & Wu, C. C. (2009). Knowledge Management and Innovation: The Mediating Effects of Organizational Learning. In IEEE (Ed.). Hong Kong.

McAdam, R. (2004). Knowledge creation and idea generation: a critical quality perspective. *Technovation, 24*, 697-705.

Mehta, N. (2008). Successful knowledge management implementation in global software companies. *The Journal of Knowledge Management, 12*(2), 42-56.

Nonaka, I. (1994). A Dynamic Theory of Organizational Knowledge Creation. *Organization Science, 5*, 14-37.

Nonaka, I., Toyama, R., & Konno, N. (2000). SECI, Ba and Leadership: a Unified Model of Dynamic Knowledge Creation. *Long Range Planning, 33*, 5-34.

Popadiuk, S., & Choo, C. W. (2006). Innovation and knowledge creation: How are these concepts related? *international Journal of Information Management, 26*, 302-312.

Rollet, H. (Ed.). (2003). *Knowledge Management: Processes and Technologies*. Massachusetts - USA: Kluwer Academic Publishers.

Ruggles, R., & Little, R. (1997). Knowledge Management and Innovation: An initial exploration: Ernst & Young LLP.

Sandberg, B. (Ed.). (2008). *Managing and Marketing Radical Innovations: Marketing new technology*: Routledge.

Sarkar, S. (Ed.). (2007). *Innovation, Market Archetypes and Outcome: An integrated framework*. New York: Physica-Verlag.

Schneider, K. (Ed.). (2009). *Experinece and Knowledge Management in Software Engineering*. Berlin, Germany: Springer-Verlag.

Schulze, A., & Hoegl, M. (2008). Organizational knowledge creation and the generalization of new product ideas: A bihavioral approach. *Research Policy, 37*, 1742-1750.

Skarzynski, P., & Gibson, R. (Eds.). (2008). *Innovation to the Core: A Blueprint for Transforming the Way Your Company Innovates*: Harvard Business Press.

Stamm, B. V. (Ed.). (2008). *Managing innovation, design and creativity*: John Wiley & Sons Ltd.

Tushman, M. L., & O'Reilly, C. A. (Eds.). (2002). *Winning through innovation: a practical guide to leading organizational change and renewal*: Harvard Business School Press.

Urabe, K., Child, J., & Kagono, T. (Eds.). (1988). *Innovation and Management: International Comparisons*. Berlin: Walter de Gruyter & Co.

Wang, F., & Ren, A. (2006). *Knowledge Management and Process Management.* Paper presented at the IAMOT, Beijing, P.R. China.

West, M. A., & Altink, W. M. M. (1996). Innovation at Work: Individual, Group, Organizational, and Socio-historical Perspectives. *European Journal of Work and Organizational Psychology, 5*(3-11).

KNOWLEDGE MANAGEMENT IN SOFTWARE DEVELOPMENT INDUSTRY

Olga Fedotova
Dep. of Economics, Management and Industrial Engineering, University of Aveiro, Aveiro, Portugal
olgafedotova@ua.pt

Leonor Teixeira
Dep. of Economics, Management and Industrial Engineering /GOVCOPP / IEETA, University of Aveiro, Aveiro, Portugal
lteixeira@ua.pt

Helena Alvelos
Dep. of Economics, Management and Industrial Engineering /GOVCOPP, University of Aveiro, Aveiro, Portugal
helena.alvelos@ua.pt

ABSTRACT

This article contains the case study that pretends to show the implementation of knowledge management (KM) during the adoption of framework for software development (SD) process improvement – Capability Maturity Model Integrated (CMMI). A multinational software development organization (SDO) that implements best practices of CMMI was the object of the study. The results of the analysis revealed the existing interdependency between the KM process and the improvement of software development process and verified that CMMI is a successful example of combination of both mentioned approaches.

KEYWORDS: Software development, Knowledge management, KM, CMMI

1 INTRODUCTION

The software development environment is characterized as a competitive one that suffers constant changes and innovations.

Knowledge is a fundamental source of efficient competitiveness and a key resource of intelligent decision making, forecasts, projects, planning, diagnosis, analysis, evaluations and intuitive judgments (Ramanujan & Kersh, 2004).

As software development is realized by people it is based on their knowledge. The software producing company's greatest assets don't lie in the products they make but in the knowledge of the people who produce those products (Dayan & Evans, 2006). KM focuses on the individual as an expert and as a bearer of important knowledge that he/she can systematically share with an organization (Ramanujan & Kersh, 2004).

Among the principal motivations of KM application during the software development process Rus, Lindvall *et al.* (2001) define the necessities of (i) capture and share of the knowledge about different processes and products, (ii) knowledge domain, (iii) acquirement of knowledge about new technologies, (iv) knowing 'who knows what', (v) working on a distance and, (vi) share knowledge about organizational policies and practices.

This article, through case study, pretends to show the implementation of knowledge management during the adoption of CMMI, more precisely of CMMI for Development (CMMI-DEV), by one of the software development organizations, and verify the idea of Wang and Ren (2006) about knowledge management implementation in the Primavera BSS by means of software process improvement.

Primavera Business Software Solutions is a software development organization that in the moment adopts best practices of CMMI in order to reach the Defined Maturity Level of the software development process.

2 RELATED CONCEPTS

The section is aimed to clarify some concepts on which the current study is based. A brief review of Knowledge Management (Section 2.1) and description of Capability Maturity Model Integrated (Section 2.2) contribute to satisfaction of this purpose.

2.1 Knowledge Management

According to Aurum *et al.* (2008), knowledge is a broad and abstract notion. Knowledge has been defined by Schneider (2009) as the human expertise stored in a person's mind, obtained through experience and interaction with person's environment.

Other approach is presented by Nonaka (1994), distinguishing two types of knowledge: tacit/implicit and explicit knowledge. Tacit knowledge is a kind of knowledge that is stored in the people's mind in the form of memory, skills, experience, education, imagination and creativity; while the explicit knowledge exists in a form of documented processes, directives, standards or patterns and is stored in textbooks, software products and documents (Dayan & Evans, 2006; Koskinen & Vanharanta, 2002). In this context and in accordance with Aurum *et al.* (2008) the objective of KM is to transfer the implicit knowledge into explicit one, as well as to transfer explicit knowledge from individuals to groups within the organization.

Davenport and Prusak (1998) define KM as "a method that simplifies the process of sharing, distributing, creating, capturing and understanding of a company's knowledge". Dayan and Evans (2006) and Goldshtein (2004) present the KM as a discipline that aims "to maximize innovation and competitive advantage that comes from the existence of greater amount of knowledge". Silva and Neves (2003) describe the KM as a group of processes and models used in order to create, use and disseminate the knowledge within the organization.

The process approach of KM described by Wang and Ren (2006) involves knowledge acquisition, knowledge share, knowledge application and knowledge innovation. Knowledge acquisition consists in collection, documentation and classification of data and information in the knowledge library. Knowledge share supposes that the knowledge is spread through the organization and the processes are communicated (Dayan and Evans

2006). Knowledge application presumes the application of the spread knowledge to the problem's solving. Finally, knowledge innovation "is the highest condition of KM" that is based on the improvement of existing knowledge "through practice, experiment and analysis" (Wang and Ren May, 2006).

Other framework of KM process presented by Rollet (2003) includes aggregating, planning, creating, integrating, organizing, transferring, maintaining and assessing of the knowledge. Knowledge planning consists in establishing the goals for each KM effort and their periodical revision. Creating the knowledge presumes the establishment of appropriate environment that stimulates the knowledge generation. Knowledge integration is responsible for the knowledge introduction to the organization through acquisition from the external sources. Organizing the knowledge means its structuring on the basis of previous classification. Transferring the knowledge consists in its planned exchange and sharing. Knowledge maintaining includes knowledge revising, correcting, updating, refining, preserving and removing activities. Finally, knowledge assessment verifies the extent to which the knowledge targets have been reached.

Directly or indirectly, the process of KM is present in organizations, with particular emphasis on high technology organizations such as Software Development Organizations.

2.1.1 Knowledge Management in Software Development Organizations

The software development industry has a hyper competitive environment that involves dynamic and continuous innovations. For this reason, the knowledge composes a fundamental source of competitive advantage for this type of organizations. Furthermore, the software engineering tasks are rather complex and need the synchronization among all the actors involved in the process. To ensure the efficiency of the task performance, procedures and templates should be implemented with the purpose to offer the guidance. These procedures and templates should incorporate the previous experience in the field and best practices in order to guarantee their quality (Schneider 2009).

KM and software development are highly related. Thus the success of SD is often determined by the efficient introduction and implementation of KM concepts, because the SD is realized by people and is based on their expertise.

Software engineering industry, just like any other, has its own terminology standards and artifacts. Their spread and circulation are fundamental to avoid misunderstandings and create the common base of references in the organization. As KM is concerned with dissemination of knowledge, it is an irreplaceable tool to support this kind of activities. For example, new projects in SDO presume the identification of competencies of the personnel that will participate in its implementation, just as verification of knowledge gap for the further organization of training sessions. KM needs to support matching between current skills for project development and type of training necessary to satisfy the project's requirements. Besides there is a necessity of knowing '*who knows what*' in order to set '*who makes what*'.

2.2 Capability Maturity Model Integrated

The CMMI was created by the Software Engineering Institute (SEI) of Carnegie Mellon University of America on the request of US Department of Defence to satisfy the need for an integrated model that would simplify the previous Capability Maturity Models implementation and that could help to overcome the "software crisis" (Jiang, et al., 2004 ; Mutafelija & Stromberg, 2003).

According to Kupla and Jonhnson (2008), the CMMI model has two possible representations: the staged and the continuous one. The continuous representation provides more flexible approach to the CMMI implementation as it "enables the organizations to choose the focus of process improvement efforts by choosing those process areas, that best benefit the organization and its business objectives" (Chrissis, Konrad et al. 2009). Besides choosing the process area (PA), the organization decides which capability level it pretends to achieve for the correspondent PA. On the other hand, the staged representation presumes the order in the implementation of PAs, according to maturity levels, "which define the improvement path for an organization from the initial level to the optimizing level" (Chrissis, Konrad et al. 2009) and provide the foundation for further improvements (Kupla and Jonhnson 2008).

The staged representation of CMMI is the widely-adopted one. This representation was also selected by the Primavera BSS for the software process improvement. The following section provides a brief description of staged representation of CMMI, based on the vision of some authors.

2.2.1 Staged Representation of CMMI

According to Chrissis, Konrad *et al.* (2009) CMMI framework divides the company's activity into 22 Process Areas. The PA is a group of related practices in the specific area that, when implemented collectively, satisfy a set of goals that are established in order to improve the process in analysis. The CMMI's structure, represented by Kupla and Jonhnson (2008) includes the following elements: (i) 5 Maturity Levels for the case of staged representation; (ii) 22 PAs that are grouped in 4 categories: Process Management, Project Management, Engineering, and Support; (iii) Generic and Specific Goals; (iv) Generic and Specific Practices; and finally (v) sub practices, typical work products, generic practices elaborations, notes. Each PA contains a number of components that are grouped into 3 categories: required, expected, and informative (Figure II.1).

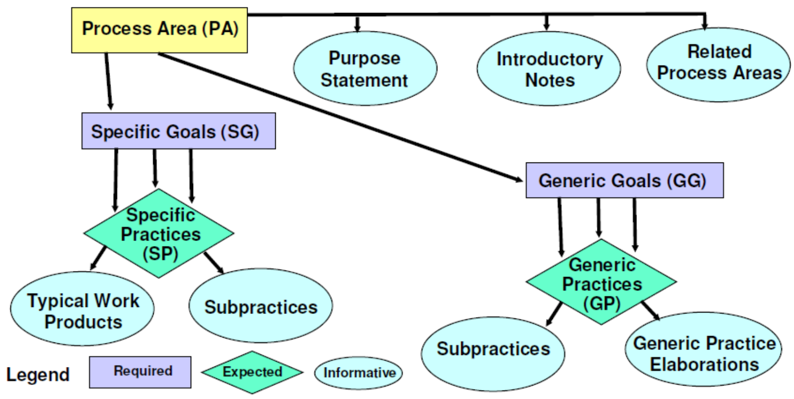

Fig.II.1 – CMMI-DEV Model Components (Buttles-Valdez, Svolou, & Valdez, March 2008).

The Specific and Generic Goals are the required components and fulfilment of these goals means the achievement of the ML to which they correspond for the PA in analysis. Specific and Generic Practices are the elements that are expected to be followed in order to achieve the Specific or Generic Goal, respectively (Kupla and Jonhnson 2008).

Sub practices, typical work products, amplifications, generic practices elaborations, and notes, are the informative components, i.e. the descriptive elements that help to understand the Specific and Generic Goals and Practices (Chrissis, Konrad et al. 2009).

Table II.1 represents the distribution of 22 PAs by existing Maturity Levels. In the Initial ML the processes are not predictable, being chaotic.

The Managed ML presumes that the processes are adherent to the organization's policy, involve relevant stakeholders, are monitored, controlled and evaluated. In order to reach this ML, each of 7 PAs correspondents to the Managed ML (Table II.1) has to achieve the Generic Goal 2 by fulfilment of the Generic Practices (from GP2.1 to the GP2.10) and achieve the correspondent Specific Goals by realization of the Specific Practices.

Table II.1 – Staged representation: PA by Maturity Level (M. Phillips, March 2007).

Level	Focus	Process Area
5 – Optimizing	*Continuous Process Improvement*	Organizing Innovation and Deployment Causal Analysis and Resolution
4 – Quantitatively Managed	*Quantitative Management*	Organizing Process Performance Quantitative Project Management
3 – Defined	*Process Standardization*	Requirements Development Technical Solution Product Integration Verification Validation Organizational Process Focus Organizational Process Definition Organizational Training Integrated Project Management Risk Management Decision Analysis and Resolution
2 – Managed	*Basic Project Management*	Requirements Management Project Planning Project Monitoring and Control Supplier Agreement Management Measurement and Analysis Process and Product Quality Assurance Configuration Management
1 – Initial		

Quality Productivity ↑

Risk Rework

The Defined ML ensures that the organization's standard processes are tailored by the guidelines for each project (Chrissis, et al., 2009). To achieve ML 3, each of 18 PAs of the ML 2 and 3 has to achieve the Generic Goal 3, besides the PAs correspondent to the Defined ML have to achieve their Specific Goals.

The Quantitatively Managed ML permits to measure the performance of the process using statistics and provides the quantitative control over this process. To achieve the ML

4, each of 20 PAs from the ML 2, ML 3 and ML 4 has to achieve the Generic Goal 3 and at least one of them has to achieve the Generic Goal 4, besides the PAs that belong to the ML 4 have to achieve their Specific Goals.

At the Optimizing ML exists the possibility to manage the continuous improvements of the process by understanding the main causes of its variation (Chrissis, et al., 2009). Similarly to the ML 4, to achieve the ML 5, all PAs (from ML 2 to ML 5) have to achieve the Generic Goal 3 and at least one of them has to achieve the ML 5, besides the PAs of the ML 5 have to achieve the correspondent Specific Goals.

According to Phillips and Kruger (2007) the evolution of KM through the maturity levels of CMMI occurs in the following manner: (i) Level 1 / Initial – *ad hoc* knowledge-based activities are not defined; (ii) Level 2 / Managed – Activities and projects are under the KM banner; (iii) Level 3 / Defined – KM shared through standardized processes; (iv) Level 4 / Quantitatively Managed – Knowledge integrated with measurements; and finally (v) Level 5 / Optimizing – KM is a self-sustaining, ongoing process.

The case study represented in the next section uses CMMI as the framework for the software process improvement because it was selected by the organization in analysis.

3 CASE STUDY: KNOWLEDGE MANAGEMENT IMPLEMENTATION BY MEANS OF SOFTWARE PROCESS IMPROVEMENT

3.1 CMMI's Incorporation within the Organizational Strategy

Before the beginning of the analysis itself, it is important to understand the role of CMMI within the Primavera BSS. The analysis of strategic map of organization provides clear description of CMMI's position in organization's strategy (Figure II.2).

This strategic map is based on the framework of Kaplan and Norton (2004). It contains two main pillars: pillar of development and pillar of profitability. The pillar of organizational development supports the left side of the map and represents the strategy for the organization's growth. It consists, basically, in the conquest of the new markets, new vertical sectors and clients fidelity. The objective 3.1 – *Conceive New Products* is the basis of this strategy. The pillar of profitability supports the right side of the map and represents the strategy for the organization's profitability growth. The achievement of such internal objectives as adoption of frameworks, automation of processes and guarantee of processes' quality contribute directly to the profitability and products' quality growth.

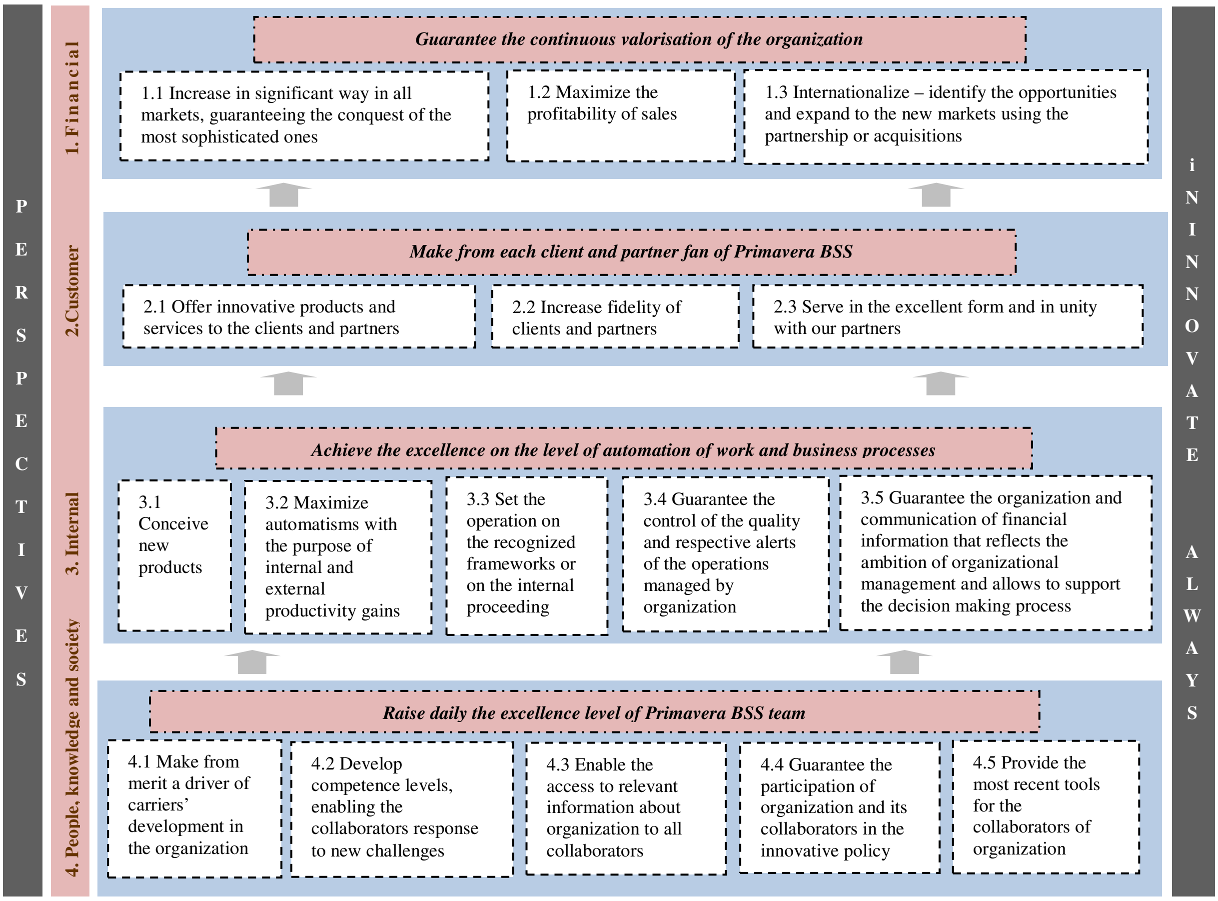

Fig.II.2 – Strategic map of Primavera BSS.

There are four main perspectives within the map: People, knowledge and society perspective; Internal perspective; Customer perspective; Financial perspective. Each of them consists of a number of objectives. Objectives of four considered perspectives are linked with each other through cause-and-effect relationships. Thus, financial objectives will be achieved only in a case of target clients' satisfaction.

The strategic objective 3.3 – *Set the operation on the recognized frameworks or on the internal proceeding* is achieved through the CMMI's implementation. The recognized frameworks, such as CMMI, SPICE, ITIL, etc. help to organizations learn faster and from the "secure" source, and to create competencies indispensable for the achievement of the customer perspective's objectives. Frameworks act like reference for the achievement of adequate quality levels of produced goods and services. Besides, they provide the common language for the whole organizational personnel.

Achievement of these objectives contributes to satisfaction of the internal perspective and is fundamental for the achievement of customer satisfaction and financial objectives.

3.2 Analysis of Improvement Infrastructure of CMMI

The further study is directed to the analysis of elaborated Improvement Infrastructure of CMMI in Primavera BSS, and search of possible connections between KM and framework for the improvement of software development process (in our case, the CMMI).

The Improvement Infrastructure is implemented in order to support the continuous software process improvement based on the organizational assets, tailoring guidelines for their standardization, personnel training and other types of information directed to enhance process performance (Figure II.3).

Organization's Set of Standard Processes contains the information about the key processes of software development and their description.

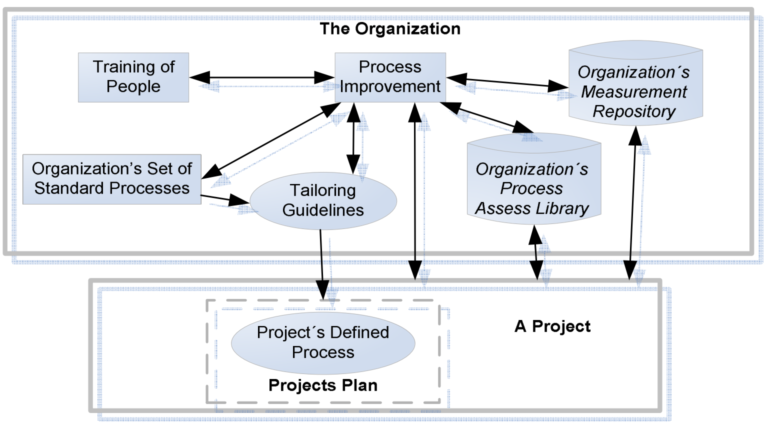

Fig.II.3 – The Improvement Infrastructure (SEI, 2009).

The software organization in analysis created the repository where the Set of Standard Processes is saved in order to fulfil the Generic Practice 2.2 – *Plan the Process* that requires the process description as the guarantee of its evolution from the chaotic and non predictable to a Managed level. All the software development processes are represented through *Software Process Engineering Meta-model* diagrams (Hauck et al., 2008), what simplifies the perception of the information. The KM emerges in the form of documentation of existing software development processes based on the best practices formed in the organization and their dissemination. It ensures the distribution of process realization through the organization, it provides the backgrounds for process improvement and serves as learning material of software development process for the organization's newcomers.

The other important component that influences the process improvement is *Training of People* (Figure II.3). The Generic Practice 2.5 of CMMI reveals the need to *Train People* in the process with the objective to help them to perform daily tasks in a better way.

There are two different types of training within the Primavera BSS classified in internal and external training. Internal training is carried out by the personnel of organization and corresponds to KM's knowledge transferring; while external training is organized by the Human Resources Department and is conducted by the external elements to Primavera BSS and answers to the needs of KM's knowledge integrating.

For each new project and according to its scope, the project competencies are determined. According to this set of competencies, the skills and carried out trainings of each element of the project team are verified. The '*SkilLand*' and '*Training*' intranet portals provide data for this analysis (Figures II.4 and II.5).

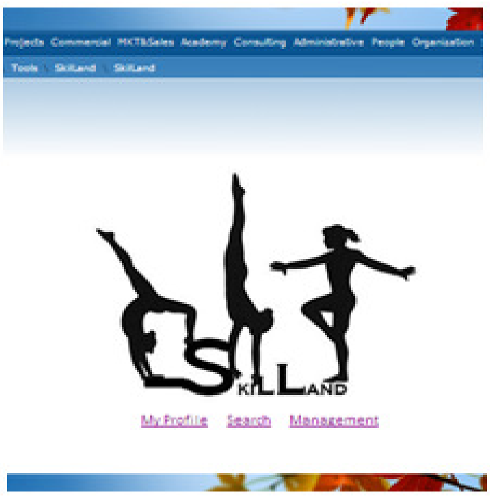

Fig.II.4 – '*SkilLand*' portal.

Fig.II.5 – '*Training*' portal.

'*SkilLand*' has the information about Primavera's employees' skills and it allows filtering the collaborators by determinate skill as well as visualizing all skills of each collaborator. '*Training*' portal contains the information about the programmed trainings of Primavera's employees and historical data of performed trainings. When the training of the collaborator is finished, the information about the obtained skills is transferred directly to the '*SkilLand*'.

Basically, '*SkilLand*' and '*Training*' are two repositories that contain information about trainings. They were personalized with the purpose to satisfy the Specific Goal 2 – *Provide Necessary Training* of Organizational Training PA, to be more exact of Specific Practice 2.2 – *Establish Training Records*, in order to register the information about the type of training, participants, skills obtained after the training and make them available at any moment.

Organizational Measurement Repository contains the data that supports the metrics elaboration. The demand of collection and storage of data is proclaimed by the Specific Practice 2.1 – *Collect Measurement Data* and Specific Practice 2.3 – *Store Data and Results*, respectively, of Measurement and Analysis PA in order to achieve Specific Goal 2 – *Provide Measurement Results*.

Primavera's Measurement Repository stores the information about the change sets of the project. In this way it provides the data necessary to create and manage the project's indicators: on scope, on time, on quality, etc. Thus, this repository can be considered as the promoter of knowledge creation as it enables the generation of new types of knowledge that result from the metrics analysis.

Organization's Process Assess Library (Figure II.3) is the collection of lessons learned, documents, metrics, best practices, etc. Specific Practice 3.4 of Organizational Process Focus PA suggests *to incorporate process related experiences into the organizational process assets*. The existence of this practice can be justified by the need of identification of strengths and weaknesses of software development process to apply them for the future projects with the objective to avoid similar failures and errors and motivate the use of the best practices to reach the success. The knowledge's capture, documentation and transfer play the primary role in the realization of this Specific Practice and a further achievement of Specific Goal 3 – *Deploy Organizational Process Assets and Incorporate Lessons Learned*.

The capture and documentation of lessons learned in Primavera BSS occurs during the project's progress, with higher emphasis at the end of the project's life cycle. The dissemination takes place on the project's closure meeting.

Finally, *Tailoring Guidelines* (Figure II.3) provide the information about the modification of the standard process, necessary to implement to respect the requirements and specifications of determinate project.

There is a bidirectional relationship between the mentioned components and process improvement (Figure II.3). Each time there is a modification or improvement of the standard process it is reflected on all the infrastructural process improvement elements, what creates an opportunity for their improvement.

4 CONCLUSIONS

The increasing competition forces organizations to look for new ways of obtaining sustainable competitive advantage. It is quite clear that information technologies do not guarantee anymore the expected competitive differentiation. Nowadays, knowledge is proclaimed to be the centre of the organizations focus in order to acquire the sustainable competitive advantage. It's all about people and their knowledge.

KM as the strategic approach becomes more and more popular among software development organizations in order to create new opportunities for this industry.

In spite of these new tendencies, the software process improvement approach continues to prosper in order to make progress advance in this environment.

According to the case study, the CMMI is a successful example of combination of both earlier mentioned approaches. From the one side, it promotes the process improvement and from the other side it cares about KM incorporation during the search for competitive advantage. The world-wide recognition of CMMI reveals the outstanding outcomes that it brings with the adoption.

It is important to emphasize that the success and the range of the benefits possible to obtain with CMMI implementation are of the proper organization responsibility. As one of the directors of the Primavera BSS referred, “CMMI is like an alpinist’s cord. If you have it, it doesn’t mean that you’ll climb the mountain”. That’s why and according to Gu and Lu (2006), the incorporation of CMMI spirit into companies and the adoption of the model to the organization’s reality and needs are fundamental for gaining so expected improvements in cost, scheduling, productivity, product quality, customer satisfaction and return of investment.

The case study has put into evidence the existing interdependency between the process improvement approach and the KM and in this way confirmed the idea of Wang and Ren (2006) exposed in the introductory section. Achievement of Generic and Specific Goals of PAs related to the earlier described Improvement Infrastructure, assumed the incorporation of basic KM concepts, as well as the realization of KM processes.

For the further study, it would be interesting to explore the reverse connection between analyzed concepts and verify if the adoption of KM fundaments contributes to improvements of development processes in software organizations.

5 REFERENCES

Aurum, A., F. Daneshgar, et al., 2008. Investigating Knowledge Management practices in software development organizations - An Australian experience. *Information and Software Technology*, Vol. 50, No. 6, pp 511-533.

Buttles-Valdez, P., A. Svolou, et al., 2008. *A Holistic Approach to Process Improvement Using the People CMM and the CMMI-DEV: Technology, Process, People, and Culture, the Holistic Quadripartite*. SEPG, Munich, Germany, Software Engineering Institute,Carneggie Mellon University.

Chrissis, M. B., M. Konrad, et al., Eds., 2009. *CMMI®: guidelines for process integration and product improvement*, Addison-Wesley.

Davenport, T.H. Prusak, L., 1998. *Working Knowledge: How Organizations Manage What They Know*, Harvard Business School Press, Boston, MA, USA.

Dayan, R. and S. Evans, 2006. KM your way to CMMI. *Journal of Knowledge Management* Vol. 10, pp 69-80.

Goldshtein, G. Y., Ed., 2004. *Strategic Innovative Management*. Taganrog, Taganrog State University of Radio Engineering

Gu, H.-Z. and H.-Y. Lu, 2006). Applying CMMI Approach to Business Process Improvement. *Electronic Commerce Studies* Vol 4, pp 97-116.

Hauck, J. C. R., C. Gresse von Wangenheim, et al., 2008. Process Reference Guides – Support for Improving Software Processes in Alignment with Reference Models and Standards. in, *Software Process Improvement*: pp 70-81.

Jiang, J. J., G. Klein, et al., 2004. An exploration of the relationship between software development process maturity and project performance. *Information and Management* Vol. 41, No. 3, pp 279 - 288

Kaplan, R. S. and D. P. Norton, Eds., 2004. *Strategy Maps: Converting Intangible Assets into Tangible Outcomes*, Harvard Business School Publishing Corporation.

Kupla, M. K. and K. A. Jonhnson, Eds., 2008. *Interpreting the CMMI: a process improvement approach*, CRC Press.

Mutafelija, B. and H. Stromberg, Eds., 2003. *Systematic Process Improvement Using ISO 9001:2000 and CMMI®*, Artech House, Inc.

Nonaka, I., 1994. A dynamic theory of organizational knowledge creation. *Organization Science* Vol. 5, pp 14–37.

Phillips, A. R. and C. J. Kruger, 2007. Minimizing dysfunctional internal competition: a strategy enabler model. *openUP*.

Phillips, M., 2007. *CMMI® Version 1.2 and Beyond*. SEPG, Pittsburgh, Software engineering Institute, Carneggie Mellon University.

Ramanujan, S. and S. Kersh, 2004. Comparison of Knowledge Management and CMM/CMMI Implementation. *Journal of American Academy of Business,*Vol. 4, pp 271-277.

Rollet, H., Ed., 2003. *Knowledge Management: Processes and Technologies*, Kluwer Academic Publishers.

Rus, I., M. Lindvall, et al., 2001. Knowledge Management in Software Engineering, DoD Data Analysis Center for Software.

Schneider, K., Ed., 2009. *Experinece and Knowledge Management in Software Engineering* Springer Berlin Heidelberg.

SEI (2009). *Introduction to CMMI*, Software Engineering Institute. Porto.

Silva, R. V. d. and A. Neves, Eds., 2003. *Gestão de Empresas na Era do Conhecimento.* Edições Sílabo.

Wang, F. and A. Ren, 2006. *Knowledge Management and Process Management*. IAMOT, Beijing, P.R. China.

SOFTWARE DEVELOPMENT EFFORT ESTIMATION: APPLICATION OF A FORMAL METHOD

Olga Fedotova
Dep. of Economics, Management and Industrial Engineering, University of Aveiro, Aveiro, Portugal
olgafedotova@ua.pt

Leonor Teixeira
Dep. of Economics, Management and Industrial Engineering /GOVCOPP / IEETA, University of Aveiro, Aveiro, Portugal
lteixeira@ua.pt

Helena Alvelos
Dep. of Economics, Management and Industrial Engineering /GOVCOPP, University of Aveiro, Aveiro, Portugal
helena.alvelos@ua.pt

ABSTRACT

Software development effort estimation is the basis for the effective project planning and scheduling as well as for the project's budget definition. This article describes the most common methods used in the software effort estimation (SEE) and presents the study of a software development organization that is implementing the software development process improvement framework Capability Maturity Model Integrated (CMMI). The technique used to obtain a formal method was based on the stepwise Multiple Linear Regression (MLR) and its results were compared to the currently used expert judgement ones. It was applied to the processes of software development and software testing. The model obtained for the testing team performed better results than the expert judgement, while for the development team no satisfactory model was deduced. Thus, there was recognized the need of identification and further collection of new variables in order to implement the model for the effort estimation for the software development team.

KEYWORDS: Software development, Effort estimation, Multiple Linear Regression

1 INTRODUCTION

Software effort estimation "is the prediction about the amount of effort required to make a software system and its duration" (Singh, Bhatia, Kaur, & Sangwan, 2008). SEE first appeared in 1950s, and since then continued to attract attention of software community specialists having the objective of developing useful models that will constructively explain the development life-cycle and accurately predict the cost of developing a software product (Boehm, et al., 2000; Jones, 2007). Since then, there were developed a lot of models for the effort and cost estimation. The diversity of these models reported in the literature can be considered as an indicator of the problem complexity since there is no unique model that completely satisfies the need for objective, fast and accurate predictions in all circumstances.

Galorath and Evan (2006) resume the steps that are generally followed to obtain the project's effort estimation to: (i) Establishment of estimation scope; (ii) Establishment of technical baseline and assumptions; (iii) Collection of data; (iv) Software Sizing; (v) Preparing of baseline estimates; (vi) Quantification of risks and their analysis; (vii) Validation and review of estimate; (viii) Creation of project plan; (ix) Documentation of estimate and lessons learned, and (x) Tracking of project throughout development.

The predictive quality of estimates determines the success of the project and helps to avoid the risks related to the cost and schedule overruns. SEE is usually required in the very beginning of the project's lifecycle, making the task of effort estimation more complex. As may be observed in Figure III.1, the error of estimation decreases as the project progresses because each subsequent project milestone brings new information that complements the existing one. In this way, as the project approaches to its conclusion, it is possible to reduce the variability of effort estimation and make more accurate predictions.

This paper describes a study carried out on Primavera Business Software Solutions (BSS) which main objective is procurement of the formal method of the effort estimation for the development and testing processes. In this way, the paper is divided in 4 sections: section 1 contains the brief introduction to the problem; section 2 provides the literature overview of the software development effort estimation techniques; section 3 presents the study in the referred organization, and finally section 4 performs the conclusions.

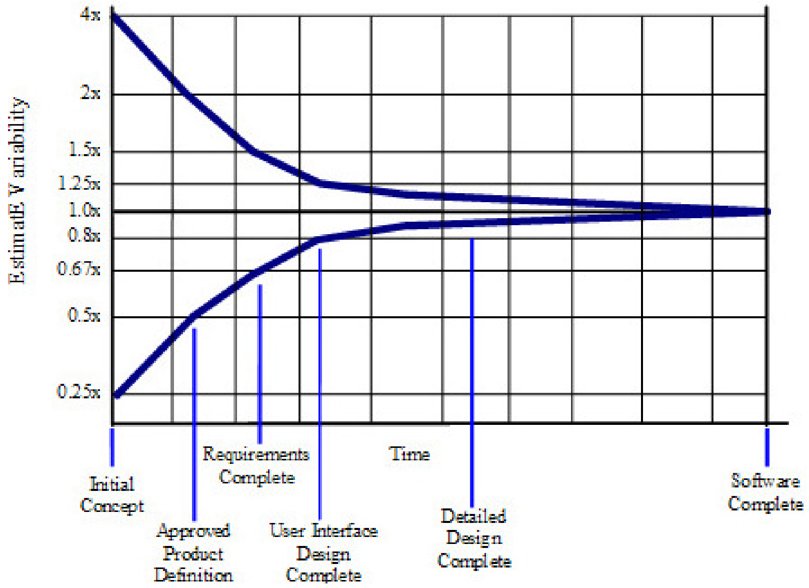

Fig.III.1 – The cone of uncertainty (McConnell, 2006).

2 SOFTWARE EFFORT ESTIMATION

This section provides literature overview of the existing classifications of software development effort estimation techniques, characterizes the most popular classification, performs the motivations and reasons for failure of the effort estimations and describes the model used to perform effort estimations in the studied software development organization.

2.1 Classification of Software Effort Estimation Techniques

The literature reports a great variety of classifications of the SEE methods.

Li, Ruhe *et al*. (2006) and Shepperd, Schofield *et al*. (1996) provide common classification of effort estimation techniques, categorizing them into expert judgement, analogy based or machine learning and algorithmic methods:

- Expert judgement effort estimation techniques are based on the person's experience and intuition (Li, et al., 2006);
- Analogy based or machine learning technique predicts the estimate from the analysis of projects with similar characteristics of the new one (Jorgensen, Indahl, & Sjoberg, 2003; Li, et al., 2006);

- Algorithmic technique is based on the mathematical models and produces effort estimation as a function of a number of variables (Leung & Fan, 2002; Li, et al., 2006).

Singh, Bhatia *et al.* (2008) give more detailed classification of effort estimation techniques than the previous one and divide them in empirical techniques, model/theory techniques, expertise techniques, regression techniques, composite techniques and machine learning techniques.

- Empirical techniques correspond to the analogy-based techniques which estimations are based on the practice and previous experience.
- Model/Theory based techniques are the algorithm bases techniques that include Function Point Analysis, SLIM, Checkpoints and COCOMO model.
- Expertise techniques are equivalent to the expert judgement when a person carries out estimation based on non-explicit and non-recoverable reasoning (Singh, et al., 2008);
- Regression based models are used to discover how the X-variables are related to Y-variables;
- Composite Techniques combine both approaches - expert judgement and project data – in a consistent manner to obtain the effort estimation (Boehm, et al., 2000; Briand & Wieczorek, 2005).

Attarzadeh and Ow (2009) and Leung & Fan (2002) give more generalized classification of effort estimation techniques dividing them into algorithmic and non-algorithmic ones. Algorithmic techniques, opposite to the non-algorithmic ones, which are soft computing based (Attarzadeh & Ow, 2009), are based on the mathematical models that are categorized as analytical and empirical ones (Leung & Fan, 2002). Empirical model derives the formula for the current project from the data available from the previous projects, while the analytical model's formula is bases on a set of global assumptions, such as the rate at which the developer solves problems and a number of available problems (Kaur, Singh, & Kahlon, 2008; Leung & Fan, 2002).

The classification of effort estimation techniques presented by Boehm, Abts *et al.* (2000) is close to the Singh, Bhatia *et al.* (2008) classification approach, adding the dynamics based techniques. Dynamics based models emphasize the dynamic character of the software project effort data and consist in application of a continuous simulation modelling methodology that detects the changes of the effort data over the duration of the project (Boehm, et al., 2000).

Laird and Brennan (2006) enrich the Li, Ruhe *et al.* (2006) and Shepperd, Schofield *et al.* (1996) classification by adding methods using benchmark data, proxy points and custom models. Models based on use of a benchmark data permit to organization that doesn't have its own database the elaboration of the formal method of effort estimation based on the existing data offered by other organization. Proxy point method decomposes the development task in components (proxies) and estimates size of each element, based on the historical data (Coleman & Verbruggen, 1998; Schoedel, 2006). Custom models opposite to all referred techniques don't impose any standard model for effort estimation, allowing modifications of formal models in order to adapt to the specific reality and needs of organization.

All mentioned classifications have a common set of techniques, which name may vary from classification to classification but the meaning maintains the same, besides some categories are particularizations of the more generic classifications. From our point of view, the presented techniques fall into one of the following classifications: (i) expert judgement; (ii) algorithmic; (iii) analogy based or machine learning; and (iv) composite techniques. These classifications can be characterized as algorithmic or non-algorithmic ones and in this way can be structured in the hierarchical manner (from the most to the less general ones). Table III.1 presents this classification and gives the respective equivalents from the earlier presented classifications.

Algorithmic models group all techniques that have mathematical basis, such as dynamics based, regression based and model/theory based techniques, while the non-algorithmic models are based on the expert judgement and analogy/machine learning techniques. Composite techniques consist on a combination of both algorithmic and non-algorithmic models what permits to this technique aggregate the advantages of both approaches.

Table III.1 – Classification and respective equivalent of the most popular techniques.

Classification		***Equivalent Technique from another Classification***	***Author of the Referenced Classification***
Non – Algorithmic Techniques	*Expert Judgement*	Expertise Technique	Singh, Bhatia *et al*. (2008)
		Expert Opinion	Laird and Brennan (2006)
		Expertise Based Technique	Boehm, Abts *et al*. (2000)
		Expert Judgement	Li, Ruhe *et al*. (2006) Shepperd, Schofield *et al*. (1996)
	Analogy base or machine learning	Empirical and Machine Learning techniques	Singh, Bhatia *et al*. (2008)
		Analogy base or machine learning	Li, Ruhe *et al*. (2006)
		Analogy	Laird and Brennan (2006)
		Learning oriented techniques	Boehm, Abts *et al*. (2000)
	Composite techniques	Composite techniques	Boehm, Abts *et al*. (2000) Singh, Bhatia *et al*. (2008)
Algorithmic Techniques	*Algorithmic Model*	Algorithmic effort estimation	Li, Ruhe *et al*. (2006) Shepperd, Schofield *et al*. (1996)
		Algorithmic Model	Attarzadeh and Ow (2009) Leung & Fan (2002) Laird and Brennan (2006)
		Dynamics based Techniques	Boehm, Abts *et al*. (2000)
		Regression Techniques	Singh, Bhatia *et al*. (2008) Boehm, Abts *et al*. (2000)
		Model/Theory technique	Singh, Bhatia *et al*. (2008)
		Model based technique	Boehm, Abts *et al*. (2000)

2.2 Advantages and Disadvantages of the Most Common Software Effort Estimation Approach

The most popular classification described in section 2.1 is the one provided by Li, Ruhe *et al*. (2006) and Shepperd, Schofield *et al*. (1996), grouping the techniques in three main categories: (i) expert judgement; (ii) analogy based or machine-learning approach; and (iii) algorithmic effort estimation. The main advantages and disadvantages of each of the categories are presented in Table III.2. This information can help in choosing which one shall be used in each specific situation.

Table III.2 – Advantages and disadvantages of the most popular classification of software effort estimation.

	Advantages	***Disadvantages***
Expert judgement	- Provides fast estimation (Bajwa, 2009) - Is useful when organization doesn't have any database (Singh, et al., 2008) - Provides estimates which are adjusted and calibrated to the past of organization by means of expert experience; - Doesn´t require any historical data; is good for unique or new projects (Galorath & Evan, 2006).	- Provides estimations that are relied on the experts experience and intuition that sometimes are questionable; factors that influence the estimation are hard to be documented (Bajwa, 2009; Singh, et al., 2008); - May not provide consistent estimation (Galorath & Evan, 2006).
Analogy based or machine-learning approach	- Is low cost, simple and relatively accurate (Hill, Thomas, & Allen, 2000); - Can employ a wide range of metrics (Koch & Mitlöhner, 2009); - Is not sensitive to the outliers; deals with poorly understood domains; can be made in the early phase of the project (Li, et al., 2006).	- Is unable to handle missing and non-quantitative data; quality of estimates relies on quality of historical data (Li, et al., 2006); - Requires database of appropriate projects (Koch & Mitlöhner, 2009); - Doesn't include adjustments related to extreme analogues and inaccurate estimations (Jorgensen, et al., 2003); - Needs analogies that match the new project characteristics (Hill, et al., 2000).
Algorithmic effort estimation	- Is objective, fast and easy to use (Galorath & Evan, 2006); - Provides relatively accurate results in a case of existence of historical database (Shepperd, et al., 1996); - Provides more objective results and can be iterated in different lifecycles (Bajwa, 2009).	- Needs to be adjusted or calibrated to the local circumstances (Shepperd, et al., 1996); - Uses size variables that are difficult to obtain in the early stages of the project; has difficulty in modelling inherent complex relationship between contributing factors; is unable to support categorical data (Attarzadeh & Ow, 2009); - Has problem of data analysis and task complexity (Vicinanza, Mukhopadhyay, & Prietula, 1991); - Is sensitive to outliers (Finnie, Wittig, & Desharnais, 1997).

Expert judgement is usually used by the organizations that don't have any database (Singh, et al., 2008), providing rather fast estimations adjusted to the past of organization. Galorath and Evan (2006) recommend this method of effort estimation for new or unique projects which characteristics don´t fall into the pattern of the past projects. Among the principal disadvantages of this method Singh, Bhatia *et al.* (2008) and Bajwa (2009) mention difficulty of extraction of factors that influence the estimation and total dependency of results accuracy on the expert experience and intuition that sometimes are questionable.

Analogy based or machine learning approach is distinguished by its low cost, simplicity and relative accuracy in a case of existence of reliable database (Hill, et al., 2000). This method can be applied on the early phases of the project's lifecycle (Li, et al., 2006), employing a great variety of metrics (Koch & Mitlöhner, 2009). Besides analogy based or machine learning approaches are not sensitive to the outliers presence and can deal with poorly understood domains (Li, et al., 2006). The main weakness of this model lies in the need of database with appropriate projects similar to the new one to realize the effort prediction of high quality (Hill, et al., 2000; Koch & Mitlöhner, 2009). Analogy based or machine learning techniques are unable to handle missing and non-quantitative data (Li, et al., 2006) and don't make adjustments related to extreme analogues and inaccurate estimations (Jorgensen, et al., 2003).

Galorath and Evan (2006) characterize algorithmic effort estimation as objective, fast and easy to use. Estimations based on this method are relatively accurate in a case of existence of historical database (Shepperd, et al., 1996). Algorithmic effort estimation, opposite to the analogy or machine learning technique, is sensitive to the outliers what may influence the quality of final results (Finnie, et al., 1997). As algorithmic effort estimation is based on the software size measure variables (such as lines of code, function points, number of functions, modules or program features required) that are normally available only in the end of the project, this type of estimations has a difficulty in being applied in the project's early stages (Attarzadeh & Ow, 2009). Algorithmic effort estimation methods, opposite to the expert judgement, need to be calibrated or adjusted to the local circumstances (Shepperd, et al., 1996) and are unable to support categorical data (Attarzadeh & Ow, 2009).

2.3 Main Motivation and Obstacles of Formal Effort Estimation Models Application

The ability to deliver the software on time, within the budget and with the expected functionalities and quality is a challenge for all software development organizations. Inaccurate estimations in software development industry is one of the most serious problems that cause the software projects failure. Both under and over estimations have negative impact on projects' results. While underestimation causes schedule delays and cost overruns that subsequently reduce the quality of end products, overestimations may lead to the loss of potential customers and partners, as well as to the inefficient distribution of the resources.

In spite of a great variety of models for software development effort estimation there is no unique method that would represent more accurate and precise results than the other ones for all projects (Finnie, et al., 1997; Singh, et al., 2008). There is no consensus in literature about the effectiveness of one or another software effort estimation method. For example, Li, Ruhe *et al*. (2006) state that in 60% of published studies analogy-based effort estimation show better results than the other two methods (expert judgement and analogy-based model), while the review of different empirical software studies made by Jorgensen (2004) revealed that there is no substantial evidence in favour of the use of estimation models over the expert estimation method that was also identified as the most frequently applied estimation strategy.

Galorath and Evan (2006) resume the main reasons for the software estimation failures to: (i) the lack of or misuse of historical data; (ii) overoptimistic leadership or management; (iii) failure to use the estimate or (iv) failure to keep the estimate current.

According to Jorgensen (2004) low popularity of software development effort models may be explained by the discomfort performed by the software development organization during the use of models that they don't fully understand and the lack of substantial evidences that the use of formal models are more accurate than the expert estimation.

The motivation of software development organization that may result in migration from the method based on expert judgement to the formal one concerns: (i) the better performance shown by models in a case of less predictive environments (Jordensen, Kirkeboen, Sjoberg, Anda, & Bratthall); (ii) independence from the experts presence and experience; and finally, (iii) satisfaction of the requirements imposed by the frameworks for software development process improvement (such as CMMI and SPICE BPG) for adoption of rationale method of effort and cost estimation in order to guarantee the evolution to more advanced capability/maturity level.

2.4 Assessment of Models' Accuracy

The major challenge of an effort estimation model consists in its capacity to produce accurate predictions. Among the main causes of inaccuracy of estimates, Jorgensen and Molokken-Ostvold (2004) refer unexpected events and overbooked tasks, change requests from the clients, problems with resource allocation, poor requirements specification, too little time spent on effort estimation work and priority on quality rather than on cost/effort accuracy. From the other hand, such factors as enlargement of the buffer in order to deal with unexpected events and requirements specification changes, experience from the previous projects, high degree of flexibility and knowledge in how to implement requirements specification, good cost control, much time spent on effort estimations positively contribute to the accuracy of estimates.

The most common accuracy predictive statistics are the mean magnitude relative error (MMRE) and the percentage relative error deviation within x (PRED(x)) (Port, Nguyen, & Menzies, 2009). Both these measures are based on the value of magnitude relative error (MRE).

MRE is a normalized measure of the discrepancy between the actual data values (in our case effort values) and the estimated values (S.G.MacDonell & Gray, 1997):

$$MRE = \frac{|Effort_{actual} - Effort_{estimated}|}{Effort_{actual}}$$

MMRE is the mean value of MRE of all observations (n) in the sample:

$$MMRE = \frac{1}{n} \times \sum_{i=1}^{n} (\text{MRE}_i)$$

The PRED(x) is defined as the average fraction of the MRE's values that are off by no more than x (Port, et al., 2009), and is calculated in the following way:

$$\text{PRED}(x) = \frac{1}{n} \times \sum_{i=1}^{n} \begin{cases} 1, \text{if MRE} \leq x \\ 0, \text{otherwise} \end{cases}$$

PRED(0,25) is used to give the percentage of estimates that were found to be within the tolerance of 25% of their actual value, but some studies also use PRED(0,20) and PRED(0,30) with little differences in results.

Conte, Dunsmore *et al*. (1986) consider the values of $MMRE \leq 0{,}25$ and $PRED(0{,}25) \geq 0{,}75$ as desirable for accurate effort model.

2.5 Multiple Linear Regression

Multiple Linear Regression belongs to an algorithmic group of techniques. According to Leung & Fan (2002) it is an empirical model that requires the data from the past projects in order to evaluate the current one. Boehm, *et al*. (2000) and Singh, *et al*. (2008) set MLR as one of the categories of effort estimation techniques (see section 2.1), that is used to find out how the independent variables (Xi) are related to the dependent variable (Y) (Anderson, Sweeney, & Williams, 2009).

MLR model is defined as:

$$Y = \beta_0 + \beta_1 X_1 + \beta_1 X_1 + ... + \beta_n X_n + \varepsilon$$

where X_1, X_2, ..., Xn are regressors; β_0 is the intercept parameter; β_1, β_2, ..., β_n are the regression coefficients; and ε is the random error component.

To assess the adequacy of the model, the coefficient of determination R^2 is used (Montgomery, Peck, & Vining, 2006). It measures the proportional reduction in variability about the mean resulting from the fitting of the multiple regression model (Freund, Wilson, & Sa, 2006).

As was already mentioned, each technique has its own specific characteristics that make it suitable to solve a particular problem. MLR technique is usually employed when (Gray & MacDonell, 1997): (i) the number of cases is significantly higher than the number of parameters to be estimated; (ii) the data has a stable behaviour; (iii) there is a small number of missing data; (iv) a small number of independent variables are sufficient, after transformations if necessary, to linearly predict the possibly transformed output variables, so as to enable an interpretable representation. According to Jorgensen (2004) regression may be used when there is a need for a simple model and analysis tool of effort estimation to support the preliminary attempts.

Application of MLR method requires verification of the associated assumptions. The major assumptions to be considered are (Freund & Wilson, 1998; Ott & Longnecker, 2010):

- Linearity – the relationship between each Xi and Y is linear, thus the model adequately describes the behaviour of data;
- The random error is an independent and normally distributed with constant variance and mean value zero.

Problems in data set or use of incorrect model may result in violation of these assumptions (Cohen, Cohen, West, & Alken, 2003).

There are several possible procedures for the selection of the independent variables to be included in the multiple linear regression model. One of them consists in the inclusion of all the independent variables that are considered relevant, while others use stepwise procedures – forward regression, backward regression and stepwise regression (Krishnaswamy, Sivakumar, & Mathirajan, 2006; Kvanli, Pavur, & Keeling, 2006). This study uses the stepwise model, which is more popular than the other ones. This method includes the independent variables one at a time (Xi), beginning with the one that has highest correlation with Y. In each step, R^2 value is evaluated and it is verified if each of the previously included variables contribute to the R^2 increase, if not it is excluded.

The next section describes the approach followed by an international software development company using a stepwise MLR method for the estimation of the effort for testing and software development teams.

3 FORMAL EFFORT ESTIMATION APPROACH: APPLICATION IN THE DEVELOPMENT AND TESTING PROCESSES

This part of the study presents the approach followed by Primavera BSS in order to implement a formal software effort estimation method.

The changes in the estimation area were originated by the adoption by organization of the framework for software development process improvement – CMMI. CMMI in order to stimulate the software development process maturity in the Project Planning process area requires the establishment of estimates for work products and tasks based on estimation rationale (Chrissis, et al., 2009).

Before the implementation of CMMI practices organization's estimates were based on the judgement of only one expert. Since it is not considered to be a valid method, there were identified two possible solutions to meet the requirements of CMMI. One of them consisted in adoption of the formal Delphi method (Leung & Fan, 2002) with participation of at least 3 area experts for the effort prediction. Another solution was the implementation of the model for the effort estimation based on the historical data of organization. Due to the more favourable costs/benefits relationship associated to the second proposal, the organization decided to proceed with it.

Project planning and further monitoring and control within the analyzed organization is made by means of change set's management. Change set (CS) is the element of work breakdown structure that is considered to be the work unit grouping a set of requirements. For this reason, project effort and cost estimation is realized in the CS level what provides possibility for more detailed cost and effort control.

Existence of two-year old database where the critical variables of occurred projects were saved and a need for a simple, objective, fast and accurate model of effort estimation to support the preliminary attempts (Magne Jorgensen, 2004) lead to explore the possibility of elaboration of software effort estimation method based on Multiple Linear Regression.

3.1 Data Analysis

The sample that was used as basis of the effort estimation method considered all closed CSs (prototyped, developed, tested and documented) from the past and current projects. Thus, there were considered 106 CSs from 13 projects of different sizes.

Taking into account the tasks of CS development, testing, prototyping and documenting, 64% of the effort is dedicated to the CS development (i.e. programming and realizing the unitary tests), 25% is spent on CS testing, while 6% and 5% are used on CS prototyping and documenting, respectively (Figure III.2).

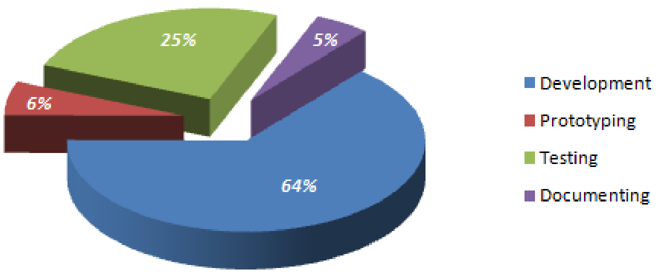

Fig.III.2 – Development effort Distribution.

The variables used to characterize the CSs sample are resumed in the Table III.3.

Table III.3 – Variables Characterization.

Variable Acronym	***Variable Description***
Dev_Eff	Effective hours spent on programming of CS and unitary tests realization.
Dev_Frc	Hours forecasted by the expert to program the CS and effectuate unitary tests.
QA_Eff	Effective hours spent on testing the CS.
QA_Frc	Hours forecasted by the expert to test the CS.
Nr_Req	Number of requirements of one CS.
Nr_CRs	Number of change requests – development tasks – per CS.
Nr_Modules	Number of modules – logic units of code - that the CS impacts.
Prot	Variable that indicates if CS will (Prot=1) or not (Prot=0) be prototyped.
Code Complexity	Ordinal variable that presents complexity of CS programming, which vary from low to high (1-3).

The descriptive statistics for all CSs presented in Table III.4 show that on average the estimated values are less than the effective values for both development and testing teams.

As may be observed in Table III.4, there is a tendency to the effort underestimation that may cause the delays on the project delivery and cost overruns. MMRE value of 0,36 for the development team shows that the forecasted values either over or under estimated the effective values. Besides the value of PRED(0,25) indicates that only 47% of the estimations are within the tolerance of 25%. Testing team performed worse results in terms of effort estimation accuracy with MMRE of 0,57 and PRED(0,25) of 37%.

Table III.4 – Descriptive Statistics of CSs.

All CSs	Development Team					Testing team				
	Dev_Eff (E)	Dev_Frc (F)	F-E	MMRE	PRED(0,25)	QA_Eff (E)	QA_Frc (F)	F-E	MMRE	PRED(0,25)
Mean	67,64	57,45	-10,19	0,36	47%	21,68	20,22	-1,47	0,57	37%
Stand. Deviation	72,49	58,46	-14,03			21,55	20,02	-1,54		
Minimum	3	3	0			1	2	1		
Maximum	363,5	280	-83,5			120	105	-15		
Median	38,25	35	-3,25			14	14	0		

3.2 Effort Estimation Model for the Testing Team

To find the effort estimation model for the testing team, the original sample of 106 CSs was reduced to 95 CSs, as there were selected only those CSs that have effective development and testing times different from zero. Variables Dev_Eff and QA_Eff were transformed to the logarithmic scale in order to have a residual distribution more approximated to the normal one.

There were obtained two models for effort estimation dedicated to the realization of testing tasks (Table III.5).

Table III.5 – Summary of the regression models for testing team.

Model Nr.	*Sample Size*	*Dependent Variables*	R^2	*MMRE Regression**	*MMRE Expert**	*PRED(0,25) Regression**	*PRED(0,25) Expert**
1.	95	ln(DEV_Eff)	62,4%	0,189	0,174	75%	73%
2.	92	ln(DEV_Eff)	71,5%	0,158	0,161	79%	74%

* MMRE Regression is calculated on the basis of the regression predicted values and effective times spent on testing (QA_Eff). MMRE Expert is calculated on the basis of QA_Eff and QA_Frc values. PRED(0,25) Expert and PRED(0,25) Regression are calculated on the basis of MMRE Expert and MMRE Regression results, respectively.

The first model was elaborated taking into account 95 CSs. To verify the existence of outliers there was elaborated the sequence chart of the studentized deleted residuals (Figure III.3), which value may vary between -2 and 2 (Pestana & Gageiro, 2003). There was revealed the existence of 3 outliers, which removal originated the sample of CSs for the second model.

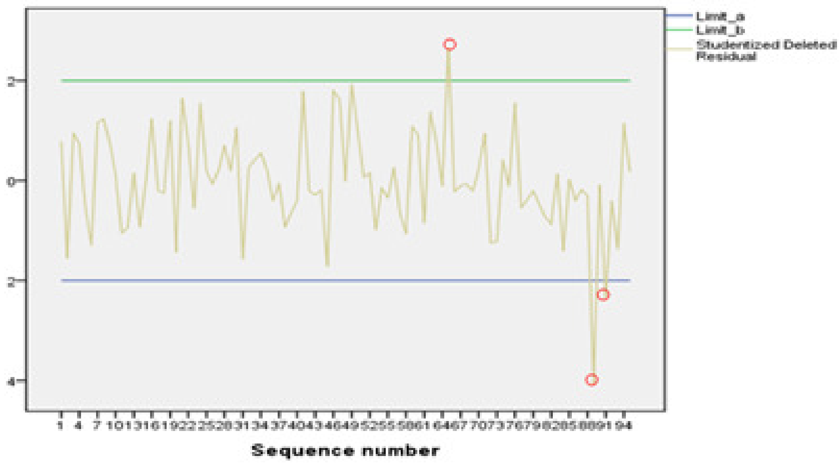

Fig.III.3 – Outliers Analysis for the Model 1**.**

In the second model the value of coefficient of multiple determination R² shows that the natural logarithm of dependent variable Dev_Eff explains 71,5% of the variation of the natural logarithm of the independent variable QA_Eff, while the first model explains only 62,4%. This improvement can be explained by the fact of outliers' elimination that resulted in the regression line's best fitting to the existing data.

Both models perform rather accurate results with values of PRED(0,25) ≥ 0,75 and MMRE ≤ 0,25. Nevertheless the second model presents better results than the first one because the results obtained with regression model are more accurate than the ones estimated by the expert as MMRE Regression < MMRE Expert and PRED(0,25) Regression > PRED(0,25) Expert .

Thus the second model was analyzed in order to verify MLR assumptions.

3.2.1 Verification of Regression Model Assumptions for Model 2

MLR is based on assumptions that errors are independent, normally distributed with constant variance and mean value zero. Validation of these assumptions is fundamental for the realization of inferences over the model's results.

Figures III.4 and III.5 and Tables III.6 and III.7 show the plots and test results of Model 2 based on the outputs from statistical tool SPSS. Kolmogorov-Smirnov and Durbin-Watson tests don't violated the assumption of the residuals' normality and independence, respectively, with the significance level of 0,489 of Kolmorov-Smirnov test (Sig>0,05) and Dublin-Watson test value of 2,139 (approximately 2).

Table III.6 – Residual Normality test.

One-Sample Kolmogorov-Smirnov Test

		Studentized Residual
N		92
Normal Parameters[a,b]	Mean	-,0001073
	Std. Deviation	1,00564300
Most Extreme Differences	Absolute	,087
	Positive	,087
	Negative	-,048
Kolmogorov-Smirnov Z		,835
Asymp. Sig. (2-tailed)		,489

a. Test distribution is Normal.

b. Calculated from data.

Table III.7 – Test of Residuals' constant variance.

Model Summary[b]

Model	Durbin-Watson
2	2,139

a. Predictors: (Constant), ln_DEV_Eff

b. Dependent Variable: ln_QA_Eff

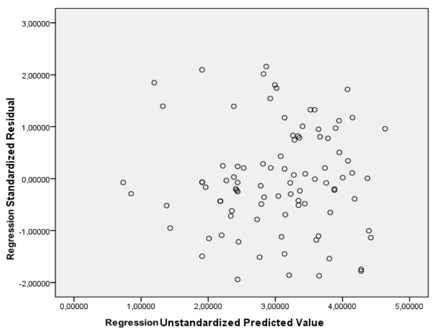

Fig.III.4 – Variance analysis with Y = ZRE[1] and X = UPRED[2].

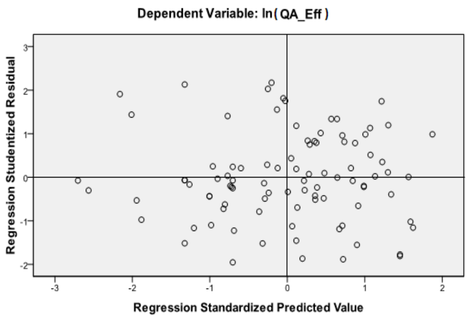

Fig.III.5 – Variance analysis with Y = SRESID[3] and X = ZPRED[4].

[1] ZRE – Regression Standardized Residual
[2] UPRED – Regression Unstandardized Predicted Value

[3] SRESID – Regression Studentized Residual
[4] ZPRED – Regression Standardized Predicted Value

Figures III.4 and III.5 are used to verify the assumption of constant variance of the residuals. As may be observed, residuals maintain approximately constant amplitude relatively to the horizontal axis and don't perform any increasing or decreasing tendency. Since there is no defined pattern in the residuals location, it may be assumed that error variance remains constant (Pestana & Gageiro, 2003).

As the effort estimation model for testing don't violate any of the MLR model's assumptions it may be further used by the organization for the testing tasks estimations. Nevertheless this model requires validation with the new projects to verify the quality of the produced results.

3.3 Effort Estimation Model for the Development Team

For the effort estimation of the software development team there were considered 106 CSs. Tables III.8 and III.9 summarize the results of the regression application.

Performed models differ one from another by the sample size and by the set of variables used to obtain the model. Variation of the sample size is justified by the elimination of the outliers. As may be observed, the higher value of R^2 corresponds to a model 4, elaborated on the basis of Prot, Code Complexity and Number of CRs variables and explain only 54,7% of the effective software development effort variation. This indicates that the existing set of variables is not enough for a satisfactory estimation of the development effort.

Table III.8 – Summary of the regression models for development team.

Model Number	Sample Size	R^2	MMRE Expert	MMRE Regression	PRED(0,25) Expert	PRED(0,25) Regression	Deviation MMRE	Deviation PRED(0,25)
1	106	0,493	0,36	1,44	44%	22%	-1,08	0,23
2	106	0,382	0,02	0,25	44%	63%	-0,23	-0,19
3	89	0,508	0,59	1,04	45%	26%	-0,45	0,19
4	89	0,547	0,16	0,20	83%	74%	-0,04	0,09
5	84	0,537	0,52	1,12	44%	40%	-0,60	0,04
6	84	0,524	0,52	1,13	44%	42%	-0,62	0,02
7	84	0,524	0,15	0,21	85%	76%	-0,06	0,08
8	84	0,358	0,15	0,22	85%	82%	-0,07	0,02
9	76	0,429	0,16	0,22	84%	72%	-0,06	0,12
10	69	0,092	0,35	1,53	46%	26%	-1,18	0,20

Table III.9 – Variables distribution per model.

Model Number	Prot	Nr_Req	Code Complexity	Nr_CRs	Nr_Modules
1	*x*		*x*	*x*	
2		*x*	*x*	*x*	*x*
3	*x*			*x*	*x*
4	*x*		*x*	*x*	
5	*x*	*x*		*x*	
6	*x*	*x*	*x*	*x*	
7	*x*	*x*	*x*		
8		*x*	*x*	*x*	
9	*x*	*x*	*x*		
10				*x*	

All regression based models, except model 2, present worse results in terms of prediction accuracy that the ones obtained with the expert opinion:

MMRE Expert < MMRE Regression

PRED(0,25) Expert > PRED(0,25) Regression

Besides, there is a great variation of MMRE and PRED(0,25) values originated by the sample size decrease or independent variables set changes. This phenomenon revealed data instability possibly originated by the absence of common reference scale for variables characterization.

Although value of PRED(0,25) Regression of model 2 is higher than of the PRED(0,25) Expert, its MMRE presents negative deviation of 25% relatively the MMRE Expert value and a set of variables explains only 38,2% of development effective effort variation.

Due to the referred analysis it was decided not to adopt any of the presented models and proceed with identification and classification of the variables that would further constitute the organization's variables database and permit creation of the formal model of effort estimation for the development team. Till that moment effort estimation procedure would be based on the formal Delphi method to guarantee the implementation of CMMI's Specific Practice 1.4 – *Determine Estimates of Effort and Cost* based on formal model. This would contribute to the achievement of Specific Goal 1 – *Establish Estimates* of Project Planning Process Area.

3.3.1 Proposal for the Variables Selection for the Characterization of the Change Set

In order to define new variables and enrich the existing set of variables that would best describe the CS, there were conducted interviews with five developers, in addition to the literature review.

One of the principal factors that determine the success of effort prediction is the size measurement. Laird and Brennan (2006) argument the importance of measuring accuracy size in the following aspects:

- Contracts assigned with customers and employees depend upon the size;
- Size shows the volume of the software;

- Effort is calculated from the size.

Koch and Mitlöhner (2009), Finnie, Wittig *et al*. (1997), Lucia, Pompella *et al*. (2005) and Hill, Thomas *et al*. (2000) among other authors distinguish size as the main factor that influences the algorithmic effort estimation approaches. Size of CS was referred by all interviewed developers as the important variable for effort estimation. One of the possible reasons for the failure of the regression-based models for the estimation in the case of the development team may consist in the absence of the size-measure variable. As may be observed from the set of variables used to classify the CS, neither of them corresponds to the CS size.

The CS Size adapted to reality of Primavera BSS can be calculated in the following manner:

$$CS\ Size = \sum_{i=1}^{n} (Requirement_i\ Complexity \times Number\ of\ Use\ Cases\ per\ Requirement_i)$$

i – number of requirement that composes CS.

Requirement complexity is the ordinal variable that varies from 1 to 3 and measures the complexity of requirement in terms of functionalities to be implemented. Number of Use Cases per requirement corresponds to the number of Use Cases that have to be specified in order to perform this requirement in the Use Case diagram.

In addition to the size variable, Hill, Thomas *et al*. (2000) refer the need to measure system complexity, personnel capabilities and experience, hardware constraints and the availability of software development tool.

According to Jones (2007) there are four key factors that have impact on software estimating methodologies: (i) the experience of personnel, (ii) the technologies used (programming languages, support tools, etc.), (iii) the development process, and (iv) the programming environment where the developer works.

All factors named by Hill, Thomas *et al*. (2000) and Jones (2007) were, in one or another way mentioned by the organization’s development specialists as the relevant ones during the software effort prediction. Thus, there were considered other five ordinal variables (which scale varies from 1 to 3 and in a case of *CS Implementation Impact* variable – from 1 to 5) to complement the CS size variable and to ensure the best characterization of the effort estimation unit – Change Set:

- *Business acquaintance* – defines the degree to which team elements are familiar with the business rules, laws, etc.:
- *CS implementation Impact* – expresses the volume of changes to make to the impacted processes;
- *Code reuse* – indicates if there is any already existing code and in which extent it will be reused;
- *Technical experience* – measures the degree of experience of the development team in using the technology and programming language;
- *Code complexity of CS* – expresses the degree of complexity of code elaboration for the CS.

Variables classification may be seen in Attachment.

The formal Delphi method incorporates the characterization of the earlier described variables in order to help in estimating the effort correspondent to each CS and, at the same time, ensures the collection of data necessary for the database creation.

4 CONCLUSIONS

The study described in this paper was carried out in a medium-sized international software development organization with the objective of obtaining the models of the effort estimation for the development and testing teams. Stepwise multiple linear regression technique was selected because of the existence of two years old historic data that could be used on the software effort estimation and due to its relative simplicity of use.

The stepwise MLR method applied to the testing team produced better estimates than those based on the single area expert. While the regression based estimates presented a mean magnitude relative error value of 0,158 and a percentage relative error deviation PRED(0,25) value of 79%, for expert estimates the values of MMRE and PRED(0,25) were, respectively, 0,161 and 74%.

Verification of the MLR assumptions didn't reveal the violation of the model's hypotheses. Still the regression based model for testing team need to be validated with new projects' data in order to verify its suitability.

The stepwise MLR applied to the data of development team didn't perform any viable model for the effort prediction. The causes of this problem can be related to the instability of variables behaviour during the inclusion of the new variables and elimination of the existing outliers and the inability of explanation of the dependent variable variation by the independent ones given by the low values of R^2.

Taking into consideration all mentioned factors, the organization's management decided not to adopt any of the deduced models for the development team effort estimation and to proceed with the relevant variables identification and classification to ensure the creation of the database necessary for the future formal method application as well as to train the development team in variables classification to avoid the data instability.

The interviews carried out with the software developers and the literature review resulted in the following set of variables to be considered for future effort estimation model: (i) CS size, (ii) business acquaintance and technical experience of the development team, (iii) CS implementation impact, (iv) code reuse and (v) code complexity of CS.

While there is not sufficient data to use the regression methods, the formal Delphi method will be applied with the referred variables, in order to satisfy the requirements of the CMMI in elaboration of project effort and cost estimations.

5 REFERENCES

Anderson, D. R., Sweeney, D. J., & Williams, T. A. (Eds.). (2009). *Statistics for Business and Economics* (10 ed.): Thomson South-Western.

Attarzadeh, I., & Ow, S. H. (2009). Software Development Effort Estimation Based on a New Fuzzy Logic Model. *International Journal of Computer Theory and Engineering, 1*(4), 1793-8201.

Bajwa, S. S. (2009). *Investigating the Nature of Relationship between Software Size and Development Effort.* Blekinge Institute of Technology, Ronneby.

Boehm, B., Abts, C., & Chulani, S. (2000). Software development cost estimation approaches - A survey. *Annal of Software Engineering, 10*, 177-205.

Chrissis, M. B., Konrad, M., & Shrum, S. (Eds.). (2009). *CMMI®: guidelines for process integration and product improvement* (2nd ed.): Addison-Wesley.

Cohen, J., Cohen, P., West, S. G., & Alken, L. S. (Eds.). (2003). *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences*. Mahwah, New Jersey: Lawrence Erlbaum Associates Inc.

Coleman, G., & Verbruggen, R. (1998). A quality software process for rapid application development. *Software Quality Journal, 7*, 107-122.

Conte, S. D., Dunsmore, H. E., & Shen, V. Y. (Eds.). (1986). *Software engineering metrics and models*: Benjamin-Cummings Publishing Co., Inc.

Finnie, G. R., Wittig, G. E., & Desharnais, J.-M. (1997). A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models. *Systems Software, 39*, 281-289.

Freund, R. J., & Wilson, W. J. (Eds.). (1998). *Regression Analysis: Statistical Modeling of a Response Variable* (1st ed.): Academic Press.

Freund, R. J., Wilson, W. J., & Sa, P. (2006). *Regression Analysis: Statistical Modeling of a Response Variable* (2nd ed.): Elsevier.

Galorath, D. D., & Evan, M. W. (Eds.). (2006). *Software Sizing, Estimation, and Risk Management: When Performance is Measured Performance Improves*: Taylor & Francis Group.

Gray, A. R., & MacDonell, S. G. (1997). A comparison of techniques for developing predictive models of software metrics. *Information and Software Technology, 39*, 425-437.

Hill, J., Thomas, L. C., & Allen, D. E. (2000). Experts' estimates of task durations in software development projects. *International Journal of Project Management, 18*, 13-21.

Jones, C. (Ed.). (2007). *Estimating Software Costs: Bringing Realism to Estimating* McGraw-Hill.

Jordensen, M., Kirkeboen, G., Sjoberg, D. I. K., Anda, B., & Bratthall, L. Human Judgement in Effort Estimation of Software Projects.

Jorgensen, M. (2004). Regression Models of Software Development Effort Estimation Accuracy and Bias. *Empirical Software Engineering, 9*, 297-341.

Jorgensen, M. (2004). A review of studies on expert estimation of software development effort. *The Journal of Systems and Software, 70*, 37-60.

Jorgensen, M., Indahl, U., & Sjoberg, D. (2003). Software effort estimation by analogy and ‘‘regression toward the mean’’. *The Journal of Systems and Software, 68*, 253–262.

Jorgensen, M., & Molokken-Ostvold, K. (2004). Reasons for Software Effort Estimation Error: Impact of Respondent Role, Information Collection Approach, and Data Analysis Methods. *IEEE Transaction on Software Engineering, 30*(12), 993-1007.

Kaur, J., Singh, S., & Kahlon, K. S. (2008, October). *Comparative Analysis of the Software Effort Estimation Models.* Paper presented at the World Academy of Science, Engineering and Technology, Auckland, New Zealand.

Koch, S., & Mitlöhner, J. J. (2009). Software project effort estimation with voting rules. *Decision Support Systems, 46*, 895–901.

Krishnaswamy, K. N., Sivakumar, A. I., & Mathirajan, M. (Eds.). (2006). *Management Research Methodology:Integration of Principles, Methods and Techniques*: Dorling Kindersley (India) Pvt. Ltd.

Kvanli, A. H., Pavur, R. J., & Keeling, K. B. (Eds.). (2006). *Concise Managerial Statistics*: Thomson Learning Inc.

Laird, L. M., & Brennan, M. C. (Eds.). (2006). *Software Measurement and Estimation: A Practical Approach*. Hoboken, New Jersey: Jonh Wiley & Sons, Inc.

Leung, H., & Fan, Z. (2002). Software Cost Estimation, *Handbook of Software Engineering and Knowledge Engineering*: Hong Kong Polytechnic University.

Li, J., Ruhe, G., Al-Emran, A., & Richter, M. M. (2006). A Flexible Method for Software Effort Estimation by Analogy. *Empirical Software Engineering, 12*(1), 65-106.

Lucia, A. D., Pompella, E., & Stefanucci, S. (2005). Assessing effort estimation models for corrective maintenance through empirical studies. *Information and Software Technology, 47*, 3-15.

McConnell, S. (Ed.). (2006). *Software Estimation: Demystifying the Black Art*. Redmond, Washington: Microsoft Press.

Montgomery, D. C., Peck, E. A., & Vining, G. G. (2006). *Introduction to Linear Regression*: John Wiley & Sons, Inc.

Ott, R. L., & Longnecker, M. (Eds.). (2010). *An Introduction to Statistical Methods and Data Analysis*: Cengage Learning Inc.

Pestana, M. H., & Gageiro, J. N. (Eds.). (2003). *Análise de Dados Para Ciências Sociais: a Complementaridade do SPSS*.

Port, D., Nguyen, V., & Menzies, T. (2009). Studies of Confidence in Software Cost Estimation Research Based on the Criterions MMRE and PRED. Retrieved 06-06, 2010, from http://menzies.us/pdf/09predmmre.pdf.

S.G.MacDonell, & Gray, A. R. (1997). *A Comparison of Modeling Techniques for Software Development Effort Prediction.* Paper presented at the International Conference on Neural Information Processing and Intelligent Information Systems, Dunedin, New Zealand.

Schoedel, R. (2006). *Proxy Based Estimation (PROBE) for Structured Query Language (SQL)*: Carnegie Mellon University.

Shepperd, M., Schofield, C., & Kitchenham, B. (1996). *Effort Estimation Using Analogy.* Paper presented at the International Conference on Software Engineering Berlin.

Singh, Y., Bhatia, P. K., Kaur, A., & Sangwan, O. (2008). *A Review of Studies on Effort Estimation Techniques of Software Development*. Paper presented at the Mathematical Techniques: Emerging Paradigms for Electronics and IT Industries.

Vicinanza, S. S., Mukhopadhyay, T., & Prietula, M. J. (1991). Software-Effort Estimation: An Exploratoy Study of Expert Performance. *Information System Research 2*, 243-262.

6 ATTACHMENT – Variables Characterization

Business acquaintance – if team elements are familiar with the business rules, laws, etc.:

1 – More than 3 years of experience;

2 – From 6 months to 3 years of experience;

3 – Between 0 and 6 months of experience;

CS implementation impact – volume of changes to make, number of processes to change:

1 – Alteration concerns only one file;

2 – Alteration concerns different files of unique functionality;

3 – Alteration concerns different files and different functionalities;

4 – Alteration concerns different files, functionalities and modules;

5 – Alteration results in a great amount of changes that are not known on the moment of estimation. The impact is considered to be maximal.

Code reuse – if the already existing code will be reused;

1 – With code reuse (more than 80% of code will be reused and the existing code has a high quality);

2 – With some code reuse (approximately 40-70% of code will be reused);

3 – Without any code reuse. This decision may be taken in 2 situations: when the code is absent and when the existing code has a low quality (evaluation made according to the code reviews criteria) and it costs less to rewrite it than to make alterations.

Technical experience – experience of development team in use of technology and in programming language;

1 – High (no changes of technology/programming language);

2 – Moderate (some changes in technology/programming language);

3 – Low (change of technology/ programming language).

Code complexity of CS – complexity of code elaboration for the determinate CS:

1 – Code has a low complexity;

2 – Code has a moderate complexity;

3 – Code has a high complexity.

GENERAL CONCLUSIONS

This dissertation, organized by three papers, intended to contribute to the software development process improvement through the application of the knowledge management (KM), innovation and effort estimation concepts in the Portuguese leader software development organization, Primavera Business Software Solutions (BSS).

The first paper, presented in Chapter I, described the innovation framework - '*I innovate!*' – adopted by Primavera BSS in order to stimulate innovation promotion. The process of innovation adopted by Primavera BSS is based on the conversion of the explicit and implicit knowledge of the individual into the organizational explicit knowledge. This relationship between the innovation process and knowledge management revealed the mutual interdependency between these concepts and confirmed that innovation, just as knowledge management, become more and more popular among the software development organizations as the strategic approaches for the new opportunities creation and organization competitiveness growth.

In spite of the innovation framework being recent and having popularity among the Primavera's collaborators as the ideas expression tool, there were planned some changes in order to positively contribute to its efficiency growth. The major changes, planned to be completed this year, deal with the expansion of the organization's innovation borders and inclusion of Primavera's partners, clients, institutions, users and sector experts, among the others, as the potential idea creators. This should lead to the innovation scope enlargement and contribute to the better understanding of the market needs.

The paper presented in Chapter II was aimed at analysing the contributions of KM to the implementation of the best practices of Capability Maturity Model Integrated (CMMI) in Primavera BSS. Analysis of improvements infrastructure of CMMI, together with the revision of the Generic and Specific Practices within the organization, put into evidence the existing interdependency between the software development process improvement and the KM and revealed that CMMI is a successful combination of those approaches. Besides, the study confirmed the idea of Wang & Ren (2006) about the realization of the KM through the improvement of the software development process.

Finally, the third paper, presented in Chapter III, was dedicated to the study of possible models for the software development effort prediction for Primavera BSS testing and development teams. To find the model for the effort estimation there was applied the Multiple Linear Regression technique. The model obtained for the effort prediction for the testing team provided better results than the previously used method based on the opinion of one area expert and passed to the validation stage.

In what concerns the development team, it was not possible to find the similar model for the effort prediction due to the lack of variables that would sufficiently explain the effort variation and due to the instability present in the available data. Analysis of the historical data revealed the need for identification and further collection of new variables to obtain the effort estimation model in the future.

Through the papers there was presented the relationship between the knowledge management, innovation, software process improvement and effort estimation concepts in the context of a software development company. All these concepts compose a chain of dependencies (Figure 4), which is oriented to the creation of competitive advantage.

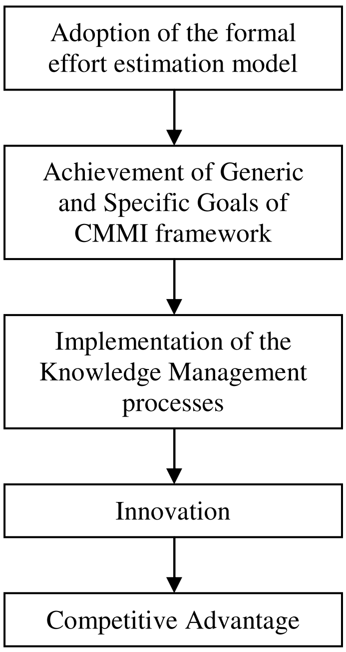

Fig.4 – Dependency chain between the effort estimation, software process improvement, KM and innovation notions for the competitive advantage creation.

With the implementation of the model for the effort estimation, as described in Chapter III, organization achieves one of the Specific Goals - *Establish Estimates* of the CMMI's framework and facilitates the improvement of the software process. As it was concluded from the second Chapter, adoption of the software process improvement framework promotes incorporation of the KM processes. From the other side, as referred in Chapter I, such KM processes as knowledge share, use, application and creation are the basis for the idea generation and innovation that are vital for the competitive advantage creation.

GENERAL REFERENCES

Allee, V. (Ed.). (1997). *The Knowledge Evolution: Expanding Organizational Intelligence*: Butterworth-Heinemann.

Boehm, B., Abts, C., & Chulani, S. (2000). Software development cost estimation approaches - A survey. *Annal of Software Engineering, 10*, 177-205.

Briand, L. C., Langley, T., & Wieczorek, I. (2000). *A Replicated Assessment and Comparison of Common Software Cost Modeling Techniques.* Paper presented at the International Conference on Software Engineering, Limerick, Ireland.

Carneiro, A. (2000). How does knowledge management influence innovation and competitiveness? *Journal of Knowledge Management, 4*(2), 87-98.

Chrissis, M. B., Konrad, M., & Shrum, S. (Eds.). (2009). *CMMI®: guidelines for process integration and product improvement* (2nd ed.): Addison-Wesley.

Darroch, J. (2005). Knowledge Management, innovation and firm performance. *Journal of Knowledge Management, 9*(3), 101-115.

Darroch, J., & McNaughton, R. (2002). Examining the Link Between Knowledge Management Practices and Types of Innovation. *Journal of Intellectual Capital, 3*(3), 210-222.

Demarest, M. (1997). Understaning Knowledge Management. *Long Range Planning, 30*, 374-384.

Gibson, D. L., Goldenson, D. R., & Kost, K. (2006). *Performance Results of CMMI®-Based Process Improvement*.

Grant, R. M. (1996). Toward a Knowledge-Based Theory of the Firm. *Strategic Management Journal, 17*, 109-122.

Jalote, P. (Ed.). (1997). *An Integrated Approach to Software Engineering*: Springer-Verlag New York Inc.

Johannessen, J.-A., Olsen, B., & Olaisen, J. (1999). Aspects of Innovation Theory Based on Knowledge-Management. *International Journal of Information Management, 19*, 121-139.

Kupla, M. K., & Jonhnson, K. A. (Eds.). (2008). *Interpreting the CMMI: a process improvement approach* (2nd ed.): CRC Press.

Kusiak, A. (2009). Innovation: A data-driven approach *International Journal of Production Economics, 122*, 440-448.

Mishra, J. K. (Ed.). (2009). *Knowledge Management: Complexity, Learning and Sustainable Innovation*: Global India Publications Pvt. Ltd.

Newell, S., Swan, J., & Robertson, M. (2000). *Contextualising Knowledge Management for Innovation*. Paper presented at the IRMA Conference.

Nonaka, I., & Takeuchi, H. (Eds.). (1995). *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*: Oxford University Press Inc.

Primavera-BSS. (2010). Primavera BSS Corporate. Retrieved 20-05, 2010, from www.primaverabss.com

Sentas, P., Angelis, L., Stamelos, I., & Bleris, G. (2005). Software productivity and effort prediction with ordinal regression. *Information and Software Technology, 47*, 17-29.

Staples, M., Niazi, M., Jeffery, R., Abrahams, A., Byatt, P., & Murphy, R. (2007). An exploratory study of why organizations do not adopt CMMI. *The Journal of Systems and Software, 80*, 883–895.

Sveiby, K. E. (Ed.). (1997). *The new organizational wealth: managing and measuring knowledge-based assets*: Berrett-Koehler Publishers, Inc.

Wang, F., & Ren, A. (2006). *Knowledge Management and Process Management.* Paper presented at the IAMOT, Beijing, P.R. China.

Wiing, K. M. (1997). Knowledge Management: An Introduction and Perspective. *The Journal of Knowledge Management, 1*(1), 6-14.