Volume 6, Issue 8, August 2016 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

A Review on Human Dynamic Conceptual Frameworks for Enhancing Quality of Software Engineering Process in Agile Environment

Pradeep Kumar Dontha
Department of Management Studies, National Institute of Technology, Trichy, Tamilnadu, India

Abstract— Conceptual frameworks for improving quality of software engineering process were presented, with emphasis on human resource management and socio cognition of employees in software industries. The extent of influence of Human Resource Management (HRM) on software engineering is the spearhead for improving software quality. This work reviews literature published in the field of software engineering process, to evaluate how the HRM and socio cognition of workforce impacts the software engineering process. At present trend most of the software firms implementing agile development environment in their software engineering process, for these software firms effective use of HRM is essential for better results. In agile environment individual human features and interactions certainly affects the quality of software. This work gives comprehensive information about conceptual frameworks, People - CMM (People Capability Maturity Model) and Pair Programming. This work also presents how the temperaments and psychological preferences of workforce are analysed, who involved in pair programming. Degree of success can be achieved through these frameworks for better software engineering results. This study also presents adaptive approaches for P-CMM and Pair Programming frameworks in agile environment.

Keywords— Software Engineering Process, Human Resource Management, P-CMM, Pair Programming, Agile environment

I. INTRODUCTION

In general Software Engineering Process potentially examined at two levels. The first level mainly deals with the technical and managerial activities that are performed during software acquisition, development, maintenance and retirement. The second level is the meta-level, which is concerned with the definition, implementation, measurement, management, change and improvement of software engineering process [1]. First level doesn't explicitly address the topics like Human Resource Management. But the meta-level of Software Engineering Process clearly compacts with HRM process.

Many theories and experiences suggested that human factors are critical for the success of software engineering process [2]. So there is a vital need of understanding and analysing human dynamics in software firms. Hence HRM is the essential substantiate for software engineering process. In general HRM process is explained as the entire range of practices and processes for managing people in the organization. HRM system is an umbrella term for integrating employee staffing, compensation-benefits, designing the work, talent management and etc. [3].

With the advancement of Technology, Knowledge and Skills many software organizations adapting new techniques and methods for better execution of software engineering process. In this competitive world acceptance of any product will be defined by its quality, stability and validity, the same will be applicable in software engineering. Hence for the better enhancement of quality in software engineering a proper and effective HRM Process is necessary. Software engineering practices widely involves humans under different roles such as Senior Project Managers, Project Managers, Business Analysts, Developers, and Testers etc. [4]. Since Software engineering process involves with massive amount of human participation, therefore it is necessary to understand their Knowledge, Skills, Personality types, Temperaments, Behaviour etc.

Organizations which adapted agile environment put more emphasis on the people and their interactions. These Agile environment organizations gives more value to the interactions among humans over processes and tools [5]. But most of the organizations with agile environment disregard their main motto of human interactions and undervalue it. So it will become a difficult task for those organizations in managing people, in order to avoid such difficulties of organizations with agile environment an appropriate human dynamic frameworks are important.

As Human Resource factors involves in Software Engineering Process, it varies according to the size of the organizations and the methodologies of developing environment they follow. In Small organizations, the employees often face problems related to issues like spiteful conditions and relation among staff [6]. On the other hand, large organisations apply both traditional methods like waterfall model and agile processes to integrate the new ones with the existing ones. So these organizations face problems due to this integration of traditional and agile environments which results in cultural, distribution of work across multiple teams in large and complex projects [7]. But there are few organizations which follow exclusively agile environment irrespective of size of the organization, these kind of organizations has to develop their own assessment and improvement process mainly for two reasons, one reason is to assure human resource quality and the second one is to understand the personnel skills, personalities, temperaments to address problems quickly.

In order to analyse the personnel quality, skills, behaviour and nature in exclusive agile environment organizations, frameworks like People Capability Maturity Model (P-CMM) and Pair Programming are to be implemented.

P-CMM is a conceptual framework for managing the development of organization personnel who are involved with software engineering process and it is a five stage process. This P-CMM framework is a quantitative model for measuring and improving performance of employees in the organizations who involves with extreme programming and also for successful implementation of staff management. Another framework for enhancing quality of software engineering process is Pair Programming, it is a formation of pair and allocation of quality at the project level based on developer’s personalities, temperaments and behaviour. Pair programming allows a systematic analysis of human dynamics and aims in providing quality at first place. The noticeable characteristics in pair programming are different personalities, temperament on communication, knowledge management and decision making. So the results of pair programming gives a detailed picture of developer personalities, temperaments on communication and coordination with other personnel who involved with software engineering process. It was reported that Pair Programming is an adaptive pair formation/rotation process model for identification, interpretation and the effective combination of developer variations to improve pair effectiveness [8].

II. PEOPLE CAPABILITY MATURITY MODEL (P-CMM): ENSURES PERSONNEL QUALITY

Evaluation and Assessment (E&A) is an acute process for both software engineering process and managing people in the organizations [9]. In general E&A were designed in the form of levels, when these different levels are managed effectively organizations following agile environment for their software engineering process achieves apex positions when compared to their previous positions. So there is an essential necessary of E&A of human resource quality and management in software organizations. This E&A in software organizations can be implemented by People Capability Maturity Model (P-CMM), it is a five level model focuses mainly on continuous improvement and management of human resources. P-CMM is a people management model, it was released in the year 1995 by CMMI institute and released version 2.0 in the year 2009 [10]. P-CMM helps different organizations in characterizing the maturity of their working personnel and begins a continuous workforce development with a set of priorities for improvement actions, integrate workforce development with process improvement and initiates a culture of excellence.

P-CMM is a tool that assists software organizations with agile environments to address the critical issues of human resources successfully [11]. P-CMM is a five staged model for E&A of human resources of organization and each level of P-CMM consists of various key process areas (KPAs). KPAs helps in identifying the clusters of related workforce practices. Implementation of P-CMM in software organizations especially with agile environment helps in establishing successive foundations for continuous improving human resource competencies, developing effective project teams, motivating improved performances and shaping the workforce of the organization needs to accomplish its future business plans. Each Maturity level of P-CMM is a well-defined evolutionary plateau that institutionalizes the new capabilities for developing organizations human resource personnel.

Table 1. Key Process Areas in various levels of People – Capability Maturity Model (P-CMM)

Maturity Level	**Focus**	**Key Process Areas (KPAs)**
Level 5 (Optimizing)	Continuously Improve and align personal, workgroup and organizational capability.	• Continuous capability improvement • Organizational Performance alignment • Continuous workforce innovation
Level 4 (Predictable)	Empower and Integrate workforce competencies and manage performance quantitatively.	• Competency Integration • Empowered Workgroups • Competency-Based Assets • Quantitative Performance Management • Organizational Capability Management • Mentoring
Level 3 (Defined)	Develop Workforce competencies and workgroups, and align with business strategy and objectives	• Competency analysis • Workforce Planning • Competency Development • Career development • Competency based practices • Workgroup development • Participatory Culture
Level 2 (Managed)	Managers take responsibility for managing and developing their people	• Staffing • Communication and Coordination • Work Environment • Performance Management

		• Training and Development • Compensation
Level 1 (Initial)	Workforce practices applied inconsistently	No Key Process Areas

A. P-CMM Initial Level (Maturity Level 1)

The initial level of P-CMM, concerns with the organizations which doesn't have consistent way of performing its work. Since most of the work processes are disordered and involves with lot of wastage of time. As managers doesn't have reliable scale to measure the efforts made by workforce to perform their duties in completing the project. In urgency towards overly aggressive deadlines for projects, the workforce staff begin ignoring the guidelines and making defects in the software engineering processes. It is very difficult to remove these defects as most of them are undetected and it is time consuming process involves with great amounts of cost. As a result the projects lose control of their schedule, quality, and costs.

Among low maturity software organizations (follows traditional way of software development practices) the work is chronically over committed and their results depends largely on the skills of exceptional individuals and works excessive hours than specified working hours. Executives of these organizations consider their human resources as assets and add value to their efforts.

Organisations implementing agile software development environment follows a high disciplined methodology for human resources practices. These organizations tends to retain skilled people, develop workforce practices and train responsible individuals to perform highly cooperative best practices. Agile environment organizations practices pair programming, encourages the tacit transmission of knowledge and promote continuous training. Project managers and mentors in these organizations are well prepared to perform their workforce responsibilities. And most of these organizations bypass the initial level of P-CMM.

For organizations involved with extreme programming same can be applicable as what agile environment organizations follows. As most of the extreme programming organizations are intended to improve the software quality and responsiveness to changing customer requirements. So these kind of organizations can implement the techniques which agile organizations implementing. Hence extreme programming organizations can also bypass the initial level of this P-CMM model.

B. P-CMM Managed Level (Maturity Level 2)

At the second level of maturity, Software organizations must establish a foundation on which they can position common processes across the organization. Before being able to implement many advanced practices like software development and quality assurance, management must establish a stable environment in which to perform professional work. Management must ensure that employees are not constantly rushing about pell-mell, cutting corners, making mistakes from hasty work and fighting the fires that characterize over-committed organizations

The primary objectives for the software organizations with maturity level 2 environment is to enable workforce to repeat practices they have used successfully in the past. To enable this repeatability, managers must get control of commitments and baselines. The effort to establish a repeatable capability is the effort to establish a basic management practices locally within each unit or project. Only when this management discipline is established, organizations will have a foundation on which it can deploy common practices.

Generally at managed level, software organization's attention focuses on unit-level issues of their workforce. An organization's capability of performing work is best characterised by the capability of workforce units to meet their commitments. This capability can be achieved by ensuring that people have the skills needed to perform their assigned work and by implementing the defined actions needed to improve performance.

The KPAs in this managed level focus on establishing a foundation of basic workforce practices that can be continuously improved to develop the capability of the workforce. This foundation of practices initially built within the units to inspire a discipline for managing people and to provide a supportive work environment with adequate work resources. The workforce units should balance their commitments with available resources. Capable people are recruited, selected and transitioned into assignments within the unit. Performance objectives are established for the committed work, and performance is periodically discussed to identify actions that can improve it. Individuals should develop interpersonal skills to ensure the work dependencies are coordinated effectively. The knowledge and skills required for performing assignments are identified and appropriate training and development opportunities are provided. The compensation is based on articulated strategy and is periodically adjusted to ensure equity.

Staffing is designed to establish a formal process by which committed work is matched to unit resources and qualified individuals are recruited, selected, and transitioned into assignments. Organizations which follow agile environment or Extreme Programming methods in their software engineering process, initiates knowledge intensive programmes for setting up the skill requirements at a higher level to coordinate their staff selection activities to attract developers capable to implement demand in agile software development and extreme programming environments like test driven development, behaviour driven development and pair programming etc.

Communication and Coordination KPA in managed level main motto is to establish a timely communication across the organizations and to ensure that the workforce has the skills to share information and coordinate their activities effectively. The Communication is very important in software organizations especially the organizations implementing agile development and extreme programming environments. Communication helps in establishing a social environment that supports effective interaction among the workforce, it is also a most significant of the four prized values, starting from the early phase of the development process and being implemented in most of the other practices programme development, testing etc. So communication and coordination establishes the initial basis for developing and empowering workforces and helps in empowering the depending workgroups in organizations.

Work environment main purpose is to establish and maintain physical work conditions and to provide resources that allow individuals and workforces to perform their tasks efficiently and without unnecessary distractions. For Agile software development organizations generally 2-12 persons in one large room with small cubicles are used side by side. All the project team members (Business Analysts, developers, testers etc.) work together in that room which is allotted.

Performance management main theme is to establish objectives related to committed work against which unit and individual performance can be measured, to discuss performance against these objectives, and to continuously enhance performance. In this performance management one can evaluate objective criteria against each individual unit and workforce unit combine. Skills and knowledge obtained by successful implementation of agile development process or Extreme Programming environments helps in capable boosting of workforce performances. Which also helps in in addressing the success requirement changes through user requirements in planning process of software, continuous integrations and small releases of developed software. The pair programming with continuous code reviews, faster code production and learning of both programming techniques and problem domain increases performance. Testing, minimizes the defect rates, and on-site customer providing feedback often and early are also significant factors affecting positively performance. The same effect is obtained with the simple design, common code ownership, and metaphor.

Training and Development key process area main motto is to ensure that all individuals have the skills required to perform their assignments and are provided relevant development opportunities. In Agile environment successful training needs by rotating developers in pair programming and by involving them in significant practices such as software planning, designing, developing, testing, refactoring and metaphor stages.

Compensation purpose is to provide all individuals with proper remuneration and benefits based on their contribution and value to the organization. Generally Software organizations with agile environment follows 40 work hours a week for benefiting both the developers and organization.

C. P-CMM Defined Level (Maturity Level 3)

The KPAs at Defined Level focus on establishing an organizational framework for developing the workforce. The organization identifies the knowledge, skills, and process abilities that underlie the workforce competencies needed to perform the business activities. The organization develops strategic plans for the workforce needed to accomplish current and future business objectives. Development opportunities are established for assisting individuals in improving their capability in these workforce competencies. The workforce practices implemented at level 2 are adjusted to motivate and support development in the organization workforce competencies. The process abilities defined for each workforce competency are used for tailoring defined process and establishing roles that provide the next step in workgroup environment. A participatory culture is established that enables the most effective use of organizations talent for making decisions and executing work.

Defined level mainly deals with the organizational issues, develops a culture of professionalism based on well understood workforce competencies. The main purpose of the *Competency analysis* key process area is to identify the knowledge, skills, and process abilities required to perform the organization's business activities so that they may be developed and used as a basis for workforce practices. The purpose of *Competency development* is to constantly enhance the capability of workforce to perform their assigned task and responsibilities perfectly. *Competency based practices* main purpose is to ensure that all workforce practices are based in part on developing the competencies of workforce.

Workforce Planning is to coordinate workforce activities with current and future business needs at both organizational and unit levels. It also provide responsible workforce activities in units with a reference for ensuring that they perform their responsibilities with an understanding how the unit's workforce activities contribute to the business.

Career Development key process main purpose is to ensure that individuals are provided opportunities to develop workforce competencies that enable them to achieve career objectives.

Workgroup Development deals in organizing the work around competency based process abilities.

Participatory Culture key process area mainly concerns with the culture in of the organization and strives in exploiting the full capability of the workforce for making decisions that affect the performance of business activities.

Organizations implementing agile methodologies can enhance workforce competencies by providing opportunities for individuals to identify, develop, and use their skills and knowledge involving them in the implementation of extreme programming practices and also for using skills and knowledge of its workforce as resources for developing the workforce competencies of others.

Agile teams, integrate with technical and business people with divergent backgrounds and skills, keep the most significant role in identification, development, and use of competency practices. These *competency practices* starts with pair programming that helps managers and developers to identify, develop and use available knowledge and skills. Technical competencies related to methodologies, project based knowledge and tool usage are improved by planning stage, pair programming, test-driven development, behaviour-driven development, refactoring, simple design, and common code ownership. Knowledge and skills, obtained by gradual training and successful projects, enhance organizations knowledge repository. The extreme programming process establishes a high *participatory culture* spreading the flow of information within the organizations and incorporating the knowledge of developers into decision-making activities, providing them with the opportunity to achieve *career development*. Iterative and incremental development with the small releases assist in *workforce planning*, which refers to coordination and synchronization of workforce activities with current and future business needs.

D. P-CMM Predictable Level (Maturity Level 4)

The KPAs at the Predictable Level focus mainly on exploiting the knowledge and experiences of the workforce framework developed at Defined Level. The competency-based processes used by different workforce competencies are intertwined to create integrated, multidisciplinary processes and collect some of the internal workforce activities. Individuals and workgroups quantitatively manage the competency based processes that are important for achieving their performance objectives. The organizations manages the capability of its workforce and the competency-based processes they perform. The effect of workforce practices on these capabilities is evaluated and corrective actions taken if necessary. Mentor use infrastructure provided by the organization's workforce competencies to assist individuals and workforce in developing their capability.

Competency Integration main purpose is to improve the efficiency and agility of interdependent work by integrating the process abilities of different workforce competencies. This competency integrating process is formed from integrating and interweaving different competency-based processes to achieve a seamless process based interaction among individuals possessing different workforce competencies. It also involves in analysing work to identify high leverage opportunities to integrate the processes used by different workforce competencies.

Empowered Workgroups purpose is to invest workgroups with the responsibility and authority for determining how to conduct their business activities most effectively. The main concept of this key process area usually implies that a workgroup is responsible for a "whole work process" [12].

Mentoring purpose is to transfer the lessons of greater experience in a workforce competency to improve the capability of other individuals. Mentoring activities are very important for organizations, the main purpose for mentoring is to support competency development, but the specific content to be imparted was not defined. At predicted level, mentoring and coaching activities are organized around and guided by a defined content of knowledge, skills, and process abilities to be imparted.

In agile programming environment team-based process helping work-groups to develop more cohesion, capability, and responsibility. Team-based practices, competency practices, training, and mentoring are the key process areas most benefiting from pairing. Mentoring in pair programming is a never-ending process, transferring interpersonal knowledge in a formal way [13]. Recent research studies shown that the assimilation time came down from 28 days to 13 days, the mentoring time was reduced from 32% to 25%, and the training effort was cut down by half [14]. Agile and Extreme Programming environments need developers to implement best practices at extreme levels using proven *competency-based activities* in their assignments. Managers must trust the results that developers produce and the agile organizations should preserve successful results in its repository and exploits them as organizational assets. Organizational assets can be used effectively again and again as corporate standards, increase productivity and spreading learning rapidly through organization. Mangers trusting competencies empower teams by transferring to them responsibility and authority for performing committed work. Developers define milestones for coordination, integrating the *competency-based activities* into a single process. This process, constituted from different workforce competencies, should be institutionalized by organization, which begins to manage its capability quantitatively. The performance of each unit and team should be measured enabling organizations performance to become more predictable. The integration of the people processes with business processes and measuring of the co-relations between the two will help and agile organization to mature up to this level.

E. P-CMM Optimizing level (Maturity Level 5)

The Optimizing level focus continuously on improving the organization's capability and workforce practices. Individuals continually improve the personal work processes they use in performing competency-based processes. Workgroups continuously improve their operating processes through improved integration of the personal work processes of their members. The organization evaluates and improves the alignment of performance among its individuals, workgroups, and units both with each other and with the organization's business objectives. The organization continually evaluates opportunities for improving its workforce practices through incremental adjustments or by adopting innovative workforce practices and technologies.

Continuous Capability Improvement – the main purpose of this KPA is to provide a foundation for individuals and workgroups to continuously improve their capability for performing competency-based practices.

Organizational Performance Alignment – the main concern of this KPA is to enhance the alignment performance results across individuals, workgroups, and units with organizational performance and business objectives.

Continuous Workforce Innovation - the main purpose of this KPA is to identify and evaluate improved or innovative workforce practices and technologies, and implement the most promising ones throughout the organization.

Agile practices, especially pair programming with pair rotation, help increasing the knowledge level of the individuals and subsequently of the team. As mentioned in the level 4, the knowledge enriches organizations knowledge repository providing both individuals and workgroups the ability to continuously improve their capabilities. This improvement occurs through incremental advance from the implementation of the agile programming practices. The results from measurements at level 4 and the culture of improvements established by the continuous implementation of the agile practices which can help organizations to mature up to this level.

III. PAIR PROGRAMMING – A PROMISING METHOD FOR DEVELOPING HIGH QUALITY SOFTWARE PROCESS

Generally it is considered as a method of programming where two people work together shoulder-to-shoulder at a single computer. Many researches has shown that pair programming yield better design, more compact code, and fewer defects for roughly equivalent person-hours [15-17]. From past few years pair programming emerged as a promising method for creating higher quality software in a time efficient manner. It is also a central aspect of many agile software development methods.

Many research studies has proven that socio-cognitive factors of programmers will effect pair programming. Studies has also noted that pair programmers exhibit greater confidence in their code and more enjoyment of programming process [18-20]. Positive results with pair programming have led to speculation that a collateral benefits of the practice may include improved morale and project knowledge shared efficiently across the development team in a manner that can be expected to improve productivity in subsequent development cycles [21]**.**

A. Pair Programming – Conceptual Framework

Pair programming is done by two programmers, working together at a single PC. Within the pair, work is split into two roles, known as the *driver* and the *navigator*. The driver is the person at the keyboard, responsible for the actual typing of the code being generated. The navigator is an active observer and monitor of the code being written. The driver and navigator collaborate on all aspects of the software development: design, coding, debugging, etc. They are in constant communication, asking and answering questions to each other. The two programmers may switch roles frequently in the course of a programming sessions.

The pair programming is best way of programming because two people make a better design decisions than one. This view characterizes programming a series of design decisions that are translated into code. The presence of second individual distributes the cognitive task [22] of the programming, aiding design discussion and error finding. The reasons why working in pairs is better when working than solo is

- Two individuals will have overlapping, but not identical, sets of information. When working together as a pair, sharing this increased pool of information can lead to better decision-making [23-24].
- Design collaboration affords a mutual apprenticeship, where through the collaboration each particular learns some of the technical skills and methods of their collaborator. This is the one of the reasons why pair programming takes place in rotation on a frequent basis [21]
- Collaborative designs requires the negotiation of a shared understanding and mutual orientation. This negotiation process makes explicit the cognitive process that are normally tacit when working individually [25].
- This negotiation process requires that programmers produce an account [26-27] of goals, plans, decisions and actions. This appears to lead to a more through exploration of design options. This account production, verification, and affirmation leads to increased confidence by the programmers and vets flawed design ideas earlier.

Working in pairs also has influences when design decisions are translated into code. By monitoring the coding, the navigator can look for missed cases and typographical errors. The navigator can also think ahead of the code being typed at a given moment. One way of stating this is that the navigator can consider issues that have a longer time constant that those being addressed by the driver [28].

The social understanding of pair programming fits into the broader frame of studying small teams engaged in engineering design, and our research draws upon this background and literature. In cognitive activity of pair programming the driver operates in relatively more concrete thinking of space by focusing on implementation, as opposed to the navigator, who deals with more abstract issues by focusing on higher level conceptual relationships and goals.

To understand the importance of pair programming initially there is a need to analyse solo programming. The solo programming depicts as follows in Figure 1.

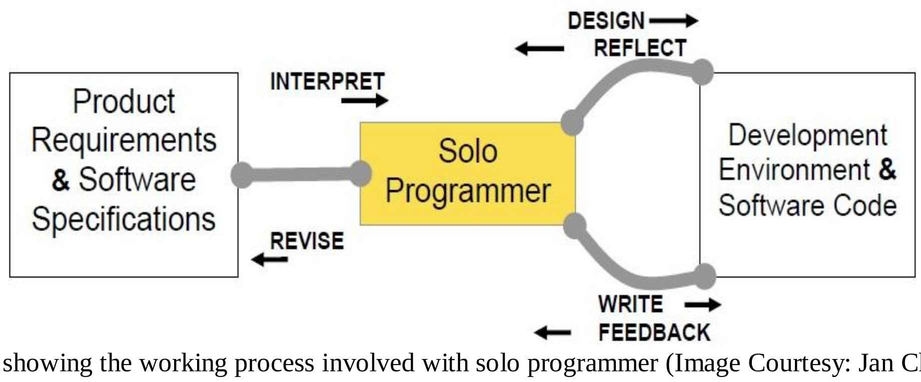
DESIGN
REFLECT
INTERPRET
Product Requirements & Software Specifications
Solo Programmer
Development Environment & Software Code
REVISE
WRITE
FEEDBACK

Figure 1. Image showing the working process involved with solo programmer (Image Courtesy: Jan Chong et al 2007)

The solo programmer, in addition to interpreting and revising the specifications, must attend to the code at many levels, ranging from high-level design and design revisions to low-level entering of program statements and understanding of debugging results. The pair programming is depicted in the Figure 2.

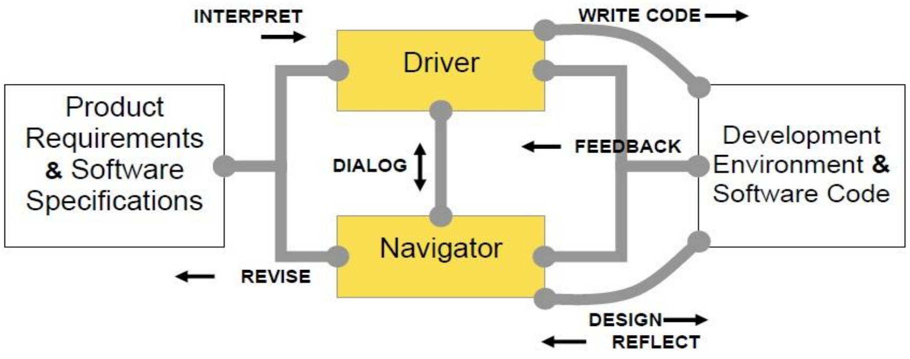
INTERPRET
WRITE CODE
Driver
Product Requirements & Software Specifications
DIALOG
FEEDBACK
Development Environment & Software Code
Navigator
REVISE
DESIGN
REFLECT

Figure 2. Image showing the process involved in pair programming (Image Courtesy: Jan Chong et al 2007)

In pair programming software development paradigm, the driver and navigator act on the specifications in tandem and develop code.

B. Human Issues in Pair Programming

From last few years pair programming has achieved the best practice in agile programming environment, but also as a standalone programming style. It is an intensively social and collaborative activity practiced by two developers working together at one machine [21]. In this Pair programming process creativity becomes a function of how developers communicate, interact and collaborate to produce results [29]. When working in pairs their personal preferences, traits, and characteristics have a strong influence on their decisions and actions.

Organizations and Managers have faced pair programming as a rough technical process [30]. In any software engineering process, there exist human factors that cannot be easily identified and understood well enough to be controlled, predicted, or manipulated. On contrary, performance and effectiveness problems always exist and must be addressed successfully. Such problems are not addressable through the known improvement approaches, as most of them focus on processes or technology, not on people. The primary role of people has been largely ignored up to now and no efforts have been devoted to increase developers' communication, collaboration, and ultimately effectiveness or to address pair problems and failures. Many researches has stated that management has much to gain from psychology to understand where and why slowdowns occur [29]. Many researches on organizational psychology has claimed that only developers with different personalities and with the same experience, if effectively combined, can minimize the communication gap [7]. This means that management must utilize processes, which first identify and understand the developers' personalities and then effectively combine their potential strengths, fostering communication and collaboration. However, one critical question that still remains to be answered is which personality types should be combined in pair formations and rotations.

1) Roles and Actions

Paired developers must succeed in their assigned roles formally or informally, either part or functional such as the role of a leader, mentor, coordinator, facilitator, innovator, analyser, tester, decision maker, negotiator, and that of a peer reviewer, to mention some of the most significant roles. To accomplish all these different roles, developers must deploy a broad set of interpersonal skills, which complement each other, ensuring effective pair interrelationship and cohesion. There are no particular guidelines for the optimal distribution of roles and tasks among the paired developers. However managers should assign roles and tasks according to the strong points of developer personalities effectively combining their talents and strengths in pair rotations.

2) Communication and Collaboration

The impact of pair programming and its effectiveness has not been empirically investigated. In general, pair programming assumes that developers with frequent, easy, face-to-face communication will find it easier to develop software, get quick feedback, and make immediate corrections in their development course. But as a software grows and pairs rotate, communication paths spread and grow, thus increasing the effort for successful communication. Therefore, collaboration and personal contact among developers must be further improved, eliminating problems and smoothening possible differences and conflicts. Developer communication, as people communication in general, is never perfect and complete depending on developers' personality preferences.

Collaboration differs from communication in the sense that it involves joint and active participation in all paired activities, especially in the creation of working software, in decision-making, and in knowledge management [31]. In pair programming many important decisions which must be made quickly and well are often left for developers. Decisions are made, but the question is what criteria are used and what the scope of the decisions is. Managers must facilitate pair decision-making, taking into account developer personality preferences and motivations, in addition to the level of knowledge and information possessed by the pair, linking successful decisions to good performance and effectiveness. The same holds for transferring and sharing knowledge. During pair programming sessions, explicit and tacit knowledge are transferred to be shared between developers. Tacit knowledge is managed first through face-to-face communication and subsequently through developer rotation, simple workable code and extensive unit tests.

3) Identifying and Understanding Personalities – Temperaments

To identify the Temperaments and Personalities widely used tools are Myers-Briggs Type Indicator (MBTI) [32] and Keirsey Temperament Sorter (KTS) [33]. The MBTI a 94 item questionnaire, focuses on four areas of opposite behaviour preferences forming 16 different personality types. It is used to quickly identify where people get their energy, how they gather information, how they make decisions, and which work style they prefer. The four preferences are *Extraverting* (E) and *Introverting* (I), *Sensing* (S) and *iNtuting* (N), *Thinking* (T) and *Feeling* (F), and *Judging* (J) and *Perceiving* (P).

Table 2. Shows the Salient Characteristics of Myers-Briggs Type Indicator (MBTI) Personality types with respect to pair programmers in agile environment.

Personality Type	**Salient Features**	**Suggested use in Pair Programming**
Extroverts	• Get energy from the outside world, experiences and interactions • Talk easily	• Suitable for interactions with users and management • May be good drivers
Introverts	• Get energy from within themselves, from internal thoughts, feelings and reflections. • Prefer finished ideas, prefer to read and think about something before start talking. • Prefer to be silent	• Might be suitable for pair programming. • Must be handled with care in meetings. • **M**ay become navigators.
Sensors	• Gather information linearly through senses. • Observe what is happening around. • Recognize the practical realities of a situation. • Take things literally and sequentially. • Concentrate on details. • Prefer tangible results clearly described.	• Probably the most capable programmers
Intuitive	• Gather information more abstractly. • See the big picture of a situation or problem. • Focus on relationships and connections between facts. • Good at seeing new possibilities and different ways of doing things.	• Probably the most capable system and application analysts.
Thinkers	• Make objective decisions • Are logical, critical and orderly. • Prefer to work with facts. • Examine carefully cause and effect of a choice or action. • Can apply problem-solving abilities.	• Suitable for making pair decisions. • Suitable for problem solving solutions.
Feelers	• Make subjective decisions. • Are driven by personal values. • Likes to understand, appreciate, and support others. • Are more people-oriented.	• Are good pair and team builders. • Are good in relations with other pairs.
Judgers	• Live in an orderly and planned way, with detailed schedules. • Prefer things decided and concluded. • Prefer to avoid last-minute stresses.	• May be good navigators. • Generally combines well with a perceiver.
Perceivers	• Live in a flexible, spontaneous way. • Rely on experience. • Leave open issues. • Explore all possibilities. • Find difficulty with decision-making. • Often relies on last minute work.	• May be good drivers. • Generally combines well with a judger.

The Keirsey Temperament Sorter (KTS), a 70-item questionnaire, classifies the 16 personality types into four temperaments types: *Artisan* (SP), *Guardian* (SJ), *Idealist* (NF), and *Rational* (NT)

Table 3. Shows the Salient Characteristics of Keirsey Temperament Sorter (KTS) Temperaments types with respect to pair programmers in agile environment.

Temperament Type	**Salient Features**	**Suggested use in Pair Programming**
Artisans (SP) (Sensing –	• Prefer concrete communication. • Prefer a cooperative path to goal	• Good and start-up persons. • Effective brainstorms.

Perceiving)	accomplishment • Possess a superior sense of timing. • Prefer practical Solutions. • Are lateral thinkers.	• May be good in decision making. • May exhibit adaptability and be innovative.
Guardians (SJ) (Sensing – Judging)	• Prefer concrete communications. • Prefer more a utilitarian approach. • Are traditionalists and stabilizers. • Prefer rules, schedules, regulations, and hierarchy. • Prefer that things remain as are.	• May be good in estimations • May be good in resource management • May be good in planning game, contracts. • Are considered very responsible succeed in assigned tasks.
Idealists (NF) (Intuitive – Feeling)	• Prefer more abstract communication • Prefer more utilitarian approach • Prefer to guide others • Excellent communicators.	• Will contribute to pair spirit and morale. • Are good in personal relationships • Are good in interaction with users and management. • May be forward and global thinkers.
Rationalists (NT) (Intuitive – Thinking)	• Prefer more abstract communication • Prefer cooperative path to accomplishment. • Are natural-born scientists, theorists, and innovators. • Possess highly valuing logic and reason. • Prefer competence and excellence	• Are good in subtask identification. • Are good in long-range plans. • Are good in analysis and design. • Are considered good in inventing and configuring.

Many researches have suggested that it is good to have various combination of pairs during pair programming like extroverts and introverts, abstract and concrete thinkers, orderly and random approaches with people who enjoy diving into details before deciding and others who decide quickly and are guided by perception. Therefore, it is up to the organizations and managers to effectively combine developer diversities in pair rotations, allowing individuals to work in roles and tasks in which they can actually succeed.

IV. ADAPTIVE APPROCHES FOR P - CMM AND PAIR PROGRAMMING FRAMEWORKS IN AGILE ENVIRONMENT

A. Adaptive Approach for P-CMM to Assess Agile Programming Organizations

The Proposed approach I suggest is an adaptive P-CMM assessment process model in the sense that the agile programming organizations assesses itself against the process areas defined in each maturity level and decides what course of action to take and how to address the improvement areas. The proposed model is divided into three stages:

Input, where people process currently used by the agile organization and the adaptive P-CMM framework entered into the process

Operation, where the assessment process takes place.

Output, where the results of the assessment process, in the form of a new improved process, are adopted by a new improved process, are adopted by the people process management task and are communicated to the organization.

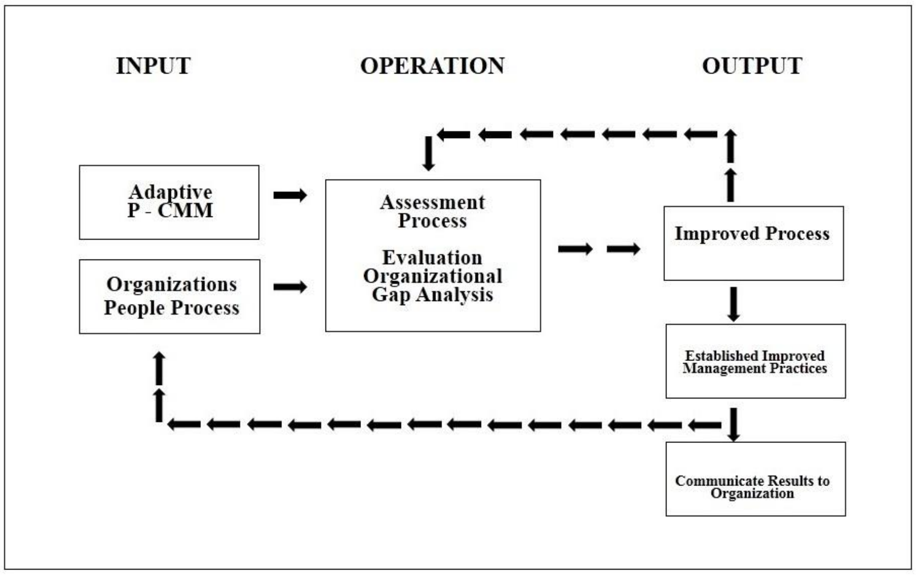
INPUT
OPERATION
OUTPUT
Adaptive P - CMM
Organizations People Process
Assessment Process
Evaluation Organizational Gap Analysis
Improved Process
Established Improved Management Practices
Communicate Results to Organization

Figure 3. Image showing the Proposed Adaptive Approach Model of P-CMM in Agile Environment

In this approach main process starts with a gap analysis [34] where organization's workforce activities are examined against P – CMM to identify the gaps or shortcomings. This kind of analysis helps the organization to measure the progress. Gap analysis can be conducted as a guided workshop session led by a qualified assessor or facilitator. Typical steps include in this gap analysis are

- An assessor or a small team of assessors consisting of managers and developers is selected and trained people in the P-CMM. After a short presentation describing the P-CMM and the purpose of survey, the program manager or facilitator assigns a specific process area and the proper evaluation questionnaire to assessors.
- Each assessor scores and comments on the process areas individually in the questionnaire, evaluating the organization against each criteria item, and determining how well the organization satisfies the described practices. Questionnaires can be filled in a group session.
- Program manager or facilitator picks up questionnaire, elaborates scores and comments analysing responses, and prioritizes results for discussions.
- Program manager or facilitator convokes a consensus meeting focussing on key areas with low scores. Meeting leads to agreement on KPAs based on overall assessment and comes to a consensus on prioritized inputs.
- Summary reports are written, describing the results for both the developer and the manager questionnaires. These reports provide team members with information about the consistency with which workforce practices are performed and about the major issues related to them. Reports provides both summary statistical data and written comments related to questions.

The assessment results are firstly incorporated into the work-force practices and secondly the improved workforce practices are established and communicated to the organization. Analytically:

- Responses are analysed and a summary presentation is delivered to the organization.
- The recommended action plans and detailed improvement activities are prioritized and incorporated into the workforce management task cycle plan to address identified areas for improvement. These steps in the assessment must be repeated in short period times at least once in a year, to keep the assessment up to date, to evaluate the progress of previously deployed assessment processes, and to use the results to feed the yearly planning cycle.

After the application of the improved process, the next step is to move into the establishing phase where a program of continuous workforce development is established. In this phase, a program of workforce development is integrated with corporate process improvement, linking together improved workforce practices with organization's workforce process. Improved workforce practices are continuously used incorporating a culture of excellence. The step is to move the communication phase where strengths, shortcomings, changes in organizational structure or processes, action plans, and detailed actions that must be taken to improve practice are communicated to the entire organization.

B. Adaptive Approach for Assessing Pair Programming in Agile Environment

In a recent field research study [35], we found out that software companies applying pair programming experienced problems due to human factors. In interviews, developers pinpointed that the most important problem they are facing is unpleasant relations with their colleagues. Besides, managers stated that such problems cannot be addressed easily because most improvement programs focus on technology, not on people. However, in general, literature and published empirical studies on pair programming do not delve issues concerning developer's personalities and temperaments and how they should be effectively combined, so as to match their potential roles and tasks. In order to obtain concrete evidences that supports or rejects the hypothesis that the combination of developers with different personalities and with the same experience can minimize communication and collaboration gaps, a research study was conducted on 84 undergraduate students. The objectives of this research study is to compare pairs comprised of mixed personalities with pairs using of the same personalities, in terms of pair effectiveness [35]. Pair effectiveness [36] was captured through: pair performance - measured by communication, velocity, productivity, and customer satisfaction (passed acceptance tests), and pair viability – measured by developers' satisfaction, knowledge acquisition, and participation (communication satisfaction ratio, nuisance ratio, and driver or navigator preference). Considering the importance of communication in pair performance, we included the communication variable in the experiment variable system. The results of the experiment have shown that there is significant difference between the two groups, indicating better performance and viability for the pairs with mixed personalities.

Based on findings of the three field studies, the results of the experiment and having the theory that considers pairs as adaptive ecosystems as framework, we propose an approach for pair formation or pair rotation process in pair programming. This model will help organizations and managers build high- performance pairs out of talented developers. It describes three main phases:

The setup phase: it includes the identification, understanding, and interpretation of the developer personalities and temperaments

The assessment phase: it includes a gap analysis and the construction or review of a set of guidelines and policies for pair formation and pair rotations

The improvement phase: it includes mini retrospectives like communication – collaboration reviews, for pair evaluation, and the establishment of the improved pair rotation process. In detail, the set of actions, which must be successively taken are

- Identify developer personalities and temperaments using MBTI and KTS tools respectively by creating personality and temperament inventories.
- Understand and interpret the impact of developer personalities and temperaments on communication and collaboration using existing personality and temperament inventories to find their strong and weak points.
- Assess existing situation analytically by performing gap analysis. First start noticing developer strengths, weakness, and oddities
  - Fit their roles and task.
  - Exhibit a steady performance
  - Take unnecessary risks, while others are conservative.
  - Construct a set of conventions and policies that might work well for them, suiting their strengths and weakness.
  - Order pair formations and pair rotations for pair programming projects, combining strengths to minimize Weaknesses, assigning the roles and tasks to developers by their strong points of their personalities.

- Monitor developer and pair performance in regular mini retrospectives, helping developers learn about themselves and how they will effectively communicate and collaborate. Retrospectives for people reviews are used in Adaptive Software Development Method [37].
- Establish improved pair formation/rotation process, communicate the results.

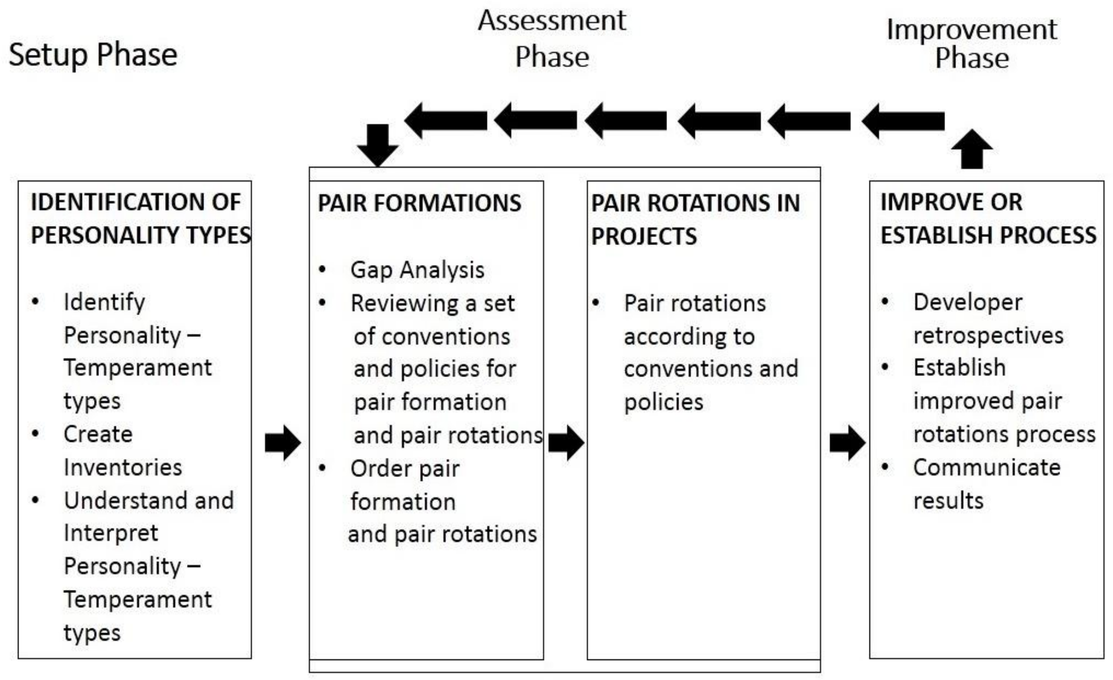
Setup Phase
Assessment Phase
Improvement Phase
IDENTIFICATION OF PERSONALITY TYPES
• Identify Personality – Temperament types
• Create Inventories
• Understand and Interpret Personality – Temperament types
PAIR FORMATIONS
• Gap Analysis
• Reviewing a set of conventions and policies for pair formation and pair rotations
• Order pair formation and pair rotations
PAIR ROTATIONS IN PROJECTS
• Pair rotations according to conventions and policies
IMPROVE OR ESTABLISH PROCESS
• Developer retrospectives
• Establish improved pair rotations process
• Communicate results

Figure 4: Image showing the Adaptive approach model for Pair formation and Pair rotations

V. CONCLUSION

A defined approaches is proposed for P-CMM assessment and improvement model for improving work force quality in agile organizations, by providing stepwise guidelines for its implementation. An agile organization's maturity from the P-CMM perspective derives from the repeatedly performed workforce practices, and the extent to which these practices have been integrated into the organizations repository. The more mature an organization, the greater its capability for attracting, developing and retaining skilled and competent employees it needs to execute its business. Agile methods, in particular extreme programming and pair programming through their repeatable practices lead to an improved work force environment with learning, training and mentoring opportunities, improving workforce competencies. Hence organizations practicing agile methods should not have problems with P-CMM and its KPAs. Agile organizations in general start with the managed level (level 2), have to make relatively limited adjustments in their workforce practices to manage other KPAs. Using measures on the performance of competency – based processes can mature an agile organization into level 4. The continuous improvement of competency – based processes, using the results of measurements, can mature an agile organization into level 5. We proposed approaches for P-CMM assessment process model for assessing agile organizations and stepwise guidelines for its implementation.

The human factors in pair programming were discussed. Considering pairs as adaptive ecosystems, we discussed how developers with different personalities and temperaments communicate, interact, and collaborate to produce results. In particular, we established the impact of developer's natural preferences and traits on the assigned roles, communication, decision-making, and knowledge management. Based on literature study of agile methods, an approach was proposed for pair formation and pair rotation process model, which identifies, interprets, and effectively combines developer variations. The proposed model can help organizations and managers to improve pair effectiveness, by matching developer's personality and temperament types to their potential roles and tasks, effectively increasing their differences in pair formations and pair rotations.

ACKNOWLEDGEMENT

I thank the large number of open source and related projects that critically facilitated this work. I thank my faculties of Department of Management Studies, National Institute of Technology Trichy. I was also grateful to Arockia Pushparaj, Senior Project Manager, Vernalis Systems Pvt Ltd Chennai for great support and assistance during this research work.

REFERENCES

[1] Software Engineering Process *by Khaled El Emam, Institute of Information Technology, National Research Council, Canada.*

[2] Human factors related challenges in software engineering: an industrial perspective, *Per Lenberg, Robert Feldt And Lars Goran Wallgren*

[3] Using Technology For Global Recruitment: Why HR/OB Scholars Need IS Knowledge? *By Elfi Furtmueller* [4] Software Engineering – ACM Digital Library *by Sommerville (2004)*

[5] Extreme Programming Explained: Embrace Change *by Beck, K. (2000)*

[6] Investigating the extreme programming system – An empirical study: Empirical Software Engineering, *Sfestos,P.., Angelis, L., & Stamelos, I. (2006a, June)*

[7] Agile Software Development. Boston: Addison – Wesley *by Cockburn, A. (2002)*

[8] Please Understand Me, Del Mar, California: Prometheus Book Company *by Keirsey, D., & Bates, M. (1984)*

[9] People Capability Maturity Model (CMU/SEI-95-MM-002 ADA300822) *by Curtis, B., Hefley, W.E., & Miller, S (1995 September)*

[10] People Capability Maturity Model (CMU/SEI-95-MM-002) *by Paulk (1995)*

[11] Software Engineering: Theory and Practice *by Shari Lawrence Pfleeger (2001)*

[12] Empowered Teams: Creating Self-Directed Work Groups That Improve Quality, Productivity, and Participation, *by Richard S.Wellins, William C. Byhamm, and Jeanne M.Wilson (1991)*

[13] Pair Programming Illuminated *by Laurie Williams and Robert R Kessler (2002)*

[14] Strengthening the case for Pair-Programming *IEEE Software by Williams, L., Kessler, R., Cunningham, W., & Jefferies, R. (2000, July/August)*

[15] A Pair Programming Experiment in a Large Computer Course *by Nicolescu, R. and R. Plummer: Romanian Journal of Information Science and Technology, 2003. 6(1-2): p. 199-216*

[16] The Costs and Benefits of Pair Programming *by Cockburn, A. and L. Williams: First International Conference on Extreme Programming and Flexible Process in Software Engineering (XP2000). 2001*

[17] Empirical Studies of Pair Programming *by Hanks, B: 2nd International Workshop on Empirical Evaluation of Agile Processes (EEAP 2003). 2003.*

[18] The Case for Collaborative Programming: Communications of the ACM *by Nosek, J.T., (1998)*

[19] Program Quality with Pair Programming in CS1 *by Hanks, B., et al: 9th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education (2004)*

[20] The Impact of Pair Programming on Student Performance, Perception and Persistence *by McDowell, C., et al. 25th Conference on Software Engineering (2003)*

[21] Extreme Programming Explained: Embrace Change *by Beck, K., (2000): Addison – Wesley*

[22] Cognition in the Wild, Cambridge, MA: The MIT Press 381 *by Hutchins, E.*

[23] Generality of a Theory of Collective Induction: Face to Face and Computer-Mediated Interaction, among Of Potential Information, and Group versus Member Choice of Evidence. Organizational Behaviour and Human Decision Process *by Laughlin, P.R., et al. (1995).*

[24] Trans-active Memory: A Contemporary Analysis of the Group Mind, in Theories of Group Behaviour *by By Wegner, D.M (1986): B.Mullen and G.R.Goethals (Editors)*

[25] Social Behaviours on XP Teams: A Comparative Study in Agile: Denver, CO *by Chong, J (2005)*

[26] Where the Action Is: The Foundations of Embodied Interaction: Cambridge, MA: The MIT Press. *By Dourish, P (2001).*

[27] Studies in Ethnomethodology: Englewood Cliffs, NJ: Prentice Hall *by Garafinkle, H. (1967)*

[28] Pair Programming: When and why it Works *by Jan Chong et al. (2003)*

[29] Extreme Programming Explained: Embrace Change (2nd Edition) *by Beck, K., (2000): Addison – Wesley*

[30] Investigating the Impact of Personality Types on Communication and Collaboration – Viability in Pair Programming – An Empirical Study: 7th International Conference on XP and Agile Processes in Software Engineering *by Sfestos, P., Stamelos, I., Angelis, L., & Deligiannis, I. (2006)*

[31] Agile Software Developments Ecosystems Boston: Addison – Wesley *by Highsmith, J. (2002)*

[32] Psychological Testing: Myers-Briggs Type Indicator *by Isabel Briggs Myers (1975)*

[33] Keirsey Temperament Sorter (KTS) *by Keirsey et al., (1984)*

[34] People Capability Maturity Model Version 2.0 (CMU/SEI-2001-MM-01) Pittsburg PA: Software Engineering Institute, CMU *by Curtis, B., Hefley, W. E., & Miller S. (2001)*

[35] Investigating the Extreme Programming System – An Empirical Study: Empirical Software Engineering *by Sfestos, P., Angelis, L., & Stamelos, I (2006a, June)*

[36] Work Teams: Applications and Effectiveness *by Sundstrom, Meuse & Futrell (1990)*

[37] Dynamic Systems Development Method: The Method in Practice *by Jennifer Stapleton (1997)*