Investigation into Cultural Aspects, Personality, and Roles of Software Project Team Configuration

by

Mohammad A. Alkandari

Thesis submitted to the faculty of
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
in
Computer Science and Applications

November 27, 2006
Blacksburg, VA

Keywords: software engineering, team configuration, culture, personality, roles

Investigation into Cultural Aspects, Personality, and Roles of Software Project Team Configuration

Mohammad A. Alkandari

(Abstract)

Managing software engineering teams in a systematic, controlled, and efficient manner often results in higher quality software. Today, with around the clock software development, software teams consist of members from geographically different locations and a variety of cultures. A few software development team compositions have been presented based on tasks, personality, and role descriptions. While these have been shown effective for understanding software teams and to some extent predicting favorable team configurations, there are no team structures or models to configure software development teams based on cultural aspects. Therefore, this thesis proposes a model for assembling software teams based on roles, personality, and cultural profiles. In particular, this research investigates how the Belbin and Myers-Briggs model, and Keirsey theories could be applied effectively to software development teams based on previous studies and analysis conducted in this study. Moreover, the study not only explores the relationship between Belbin roles and Myers-Briggs personality types, but examines how cultural differences with respect to their values and other project influences could be mapped successfully into the team profiles.

Acknowledgments

First of all, I am grateful and thankful to Allah (God) who blessed me with guidance, assistance, support, patience, and good health, as well as enabled me to write this thesis. I am also appreciative and thankful to my beloved parents, and wife, who supported and helped me all the time until I have reached where I am today.

Moreover, I am grateful and thankful to my academic advisor Dr. Shawn Bohner, especially for his advice, motivation, direction, inspiration, and encouragement that lead to the fulfillment of my thesis. I am also grateful to my committee members Dr. Deborah Tatar and Dr. James Arthur for their additional directions, supports, and valuable suggestions that helped me progress towards producing this piece of work.

Furthermore, I would like to thank Deirdre W. Jain for editing the writing of my thesis. I would like also to thank and appreciate Dr. Sallie Henry, Solomon Gifford, and Todd Stevens for their preliminary work in this area.

Last but not least, I would like to thank Kuwait University and the Embassy of the State of Kuwait at Washington DC for giving me this opportunity to pursue my graduate study in computer science. Additionally, I like to thank the computer science department at Virginia Tech including faculty members, staff, and graduate students who helped me improve my knowledge and experience in the computer science field.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Software engineering is becoming a core business process in commerce, industry, government, medicine, education, entertainment, and society at large. The software engineering field is very important today, since obtaining economical software that is both reliable and works efficiently on real machines is not an easy task. Moreover, understanding the essence of software is becoming more difficult because of its complexity, conformity, changeability, and invisibility. Consequently, much of work has to be done on analysis, evaluation, specification, design, architecture, and evolution of software to produce high quality software.

The goal of most software projects is to produce a quality software product that is in line with the requirements and resources provided. To produce a quality product, we must be mindful of the process and the requisite resources to produce the software within quality, time, and budget constraints. The most costly and variable resource for a software project manager is the software staff member. To the degree that the software team is configured with staff that is aligned well with the project goals both from the technical perspective (i.e., experience and skills) and team perspective (i.e., roles and responsibilities), the potential of a successful outcome increases. This research focuses on the team aspects of a software project. Software engineering management is the core of software development since managing software projects, processes, products, and people in a systematic, controlled, and efficient manner would produce high quality software.

John states eloquently, "human and social factors have a very strong impact on the success of software development endeavors and the resulting system (John 2005)." Software engineering (i.e., requirements analysis, design, planning and scheduling, risk analysis, and quality assurance) entails people in key roles that often determine the success of a project. Therefore, selecting the right people for these roles, based on their personalities (and perhaps their cultures), to form software teams is not an easy task. Project managers must have the appropriate skills to communicate, coordinate, and direct team members. Team members should possess the skills, knowledge, and experience to support the different roles of each software development activity in order to effectively deliver a high quality product.

1.1 Motivation

Software engineering management focuses on product, process, project, and people. *Product* is the software that is needed to be built; *process* is the set of software engineering activities required to produce a software product; and *project* is the work tasks allocated to relevant resources to deliver a bona fide product. However, *people* are the most important element of a successful project since they are responsible for developing high quality software (Pressman 2005). In general, most researchers have spent a lot of time on improving requirements engineering tools, design models, architecture designs, component level designs, user interface designs, coding principles, software testing strategies, software metrics, validation and verification techniques, and quality assurance principles. These all help the software engineer more effectively produce software. However, on the human side of the equation, how do the software team configuration and the interactions between team members impact the outcome of software projects? There are some models that help with this aspect of software project management (Belbin 1981, 1996; Myers 1998).

It is obvious that software engineering studies consist of various techniques that could result in developing high quality and low cost software, with respect to the logical methods from which those techniques were developed. However, people should follow and implement these techniques efficiently. In other words, people need skills, knowledge, and experience in order to apply the different software engineering techniques in an effective manner. For example, some roles require a high level of communication among team members, and others require a high level of creativity. In general, people vary according to their cognitive and personality styles; behaviors; experiences; abilities; education and training; motivation; management; and communications (Nash 1988). Additionally, each software engineering role requires a particular personality type and every team member belongs to a different culture which, in turn, has its own characteristics. More specifically, different cultures think and communicate differently, in which teams with these varied cultures must be carefully selected. In other words, the people factor is so important that software engineers should take into consideration how to recruit, select, train, organize, and assign each task to the right person.

The main objective of structuring a software team is to lead, organize, collaborate, motivate, and create good ideas. For example, senior managers should have enough experience to define the business issues that often have significant effects on the project. Technical managers should have enough experience to plan, motivate, and control the practitioners who create the software. Team leaders should have the ability to encourage technical people to produce to their best abilities, and encourage team members to create, as well as feel creative, even if they must work within constraints that are established for a particular software product or application. Simultaneously, team members should have trust in one another in order to communicate successfully, and the distribution of skills among team members should be suitable to the problem. Team members should have the ability to communicate with a customer in order to get the right requirements; translate customer's requirements into a designed prototype; do programming; do testing; and deliver the final product successfully to the customer. Project managers should have the ability to perform risk analysis; do project estimation; solve critical or unexpected problems; and find possible alternatives (Pressman 2005). It is clear that software projects need individuals with different personalities and capabilities in order to perform the various roles. Forming the right team would positively affect the project's outcome by avoiding delay issues or cost overruns, as well as unexpected or critical situations.

1.2 Problem Statement

Most of the software engineering studies from the last decade have not focused on the human side of software engineering – rather, they have predominantly concentrated on the analysis, design, and construction techniques necessary to produce software (John 2005). There has been substantial research progress outside of the computer science domain that aids teambuilding, promotes teamwork, and fosters communication, coordination, and collaboration. However, no work has presented a complete model by which software teams could be successfully assembled based on personality characteristics (Gifford 2003). Moreover, no work has proposed the relationship between personality profiles and cultural differences. Also, a small number of studies have addressed the personality composition of team members within a software project context (Gorla 2004). Thus, this research studies previous models that have been developed for forming teams and managing the general population, and discovers how those models could be applied effectively to software engineering team configuration based on team roles, personality, and cultural differences. Furthermore, it considers other factors that could have a significant effect on team dynamics such as knowledge, experience, and skills. This would help software managers organize and distribute effective team members into groups, with respect to the type of the task, in order to perform high quality software work, as well as assist project managers to coordinate team members more effectively to achieve successful communication. In addition, this helps project managers combat the complexity, confusion, and significant difficulties within the software teams in order to accomplish the required tasks.

1.3 Research Goals

The main goals of this current research are:

Goal 1: To study the different models that have been developed for general teambuilding.

Goal 2: To investigate how those models could be applied to software development teams based on roles of a project manager, team leader, requirements engineer, architect, and designer.

Goal 3: To examine personality factors as they pertain to team formulation

Goal 4: To investigate cultural factors as they could pertain to team formulation.

Goal 5: To propose a model that builds on previous research with roles and personality, and introduces cultural factors as a key determinant for assembling effective teams.

1.4 Research Approach

This research compares relevant teambuilding models for general projects for the purpose of seeing how they support software project teams. Case studies are examined to determine their key components, benefits, and claims with respect to team formation. This research examines some of the quantitative and qualitative aspects of each model to understand how these models could be used as the basis for a model for software project team decisions. It shows how team building models and the roles of project manager, team leader, requirement engineer, architect, and designer require different types of personalities as well as several levels of knowledge, skills, and experience. Ultimately, this study explores the relationship between personality and cultural differences, and proposes how cultural aspects may have significant effects on team formation (See Figure 1.1).

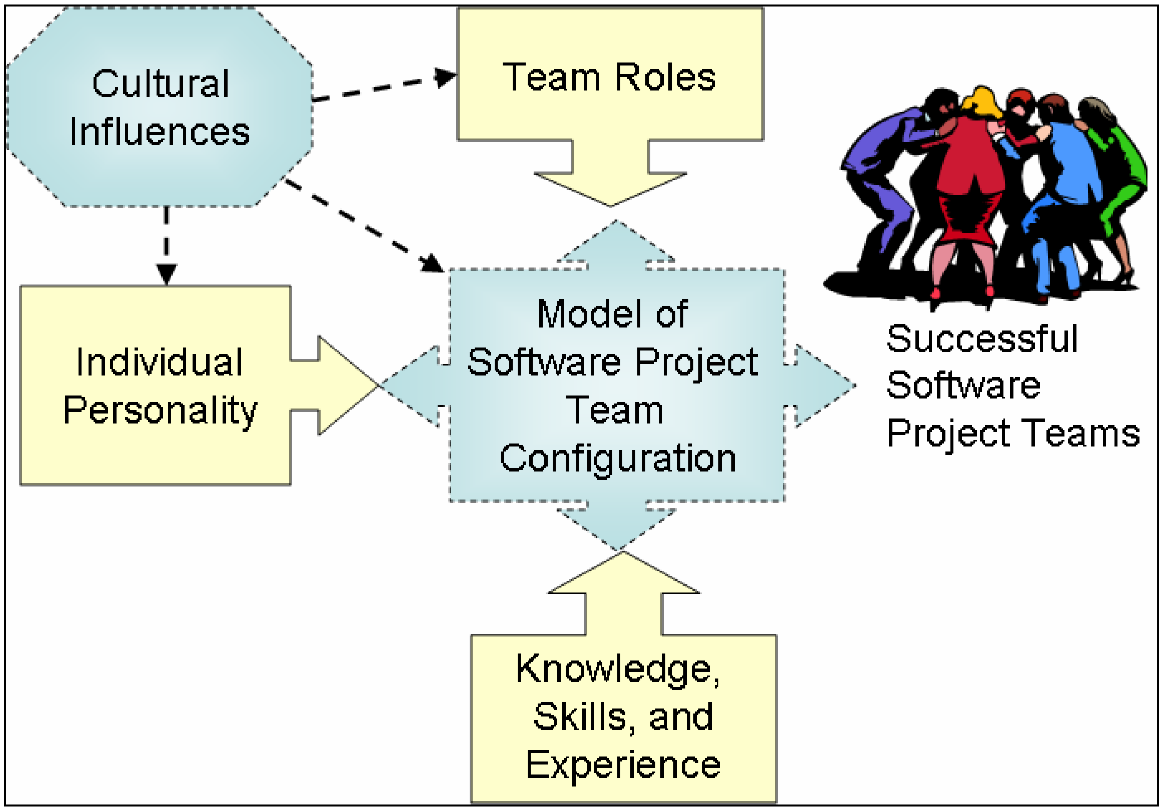
Cultural Influences
Team Roles
Individual Personality
Model of Software Project Team Configuration
Successful Software Project Teams
Knowledge, Skills, and Experience

Figure 1.1: Focus of the Research Approach

As depicted in Figure 1.1, there are three areas that have models to support project team configuration: Knowledge/Skills/Experience, Team Roles, and Individual Personality. This research endeavors to use these models coupled with a model for cultural influences to establish a viable model for configuring software project teams. Recognize that the scope is limited to software project teams and areas where culture factors are relevant. The goal is to give software project managers a model to reason about software project team configurations that are more effective.

Gifford expressed the need for research investigating individual differences in personalities with respect to their cultures as a source for helping software managers structure their teams (Gifford 2003). A study by Stevens examines the effects of roles and personality characteristics on software development team effectiveness (Stevens 1998). Belbin (Belbin 1981, 1996) suggests that a team with all clever people may not meet the projects' deadlines successfully because they waste time on debating, trying to persuade other members, neglecting important issues, as well as no coherence in the decision making. Howard (Howard 2001) adds that understanding developer personality characteristics can assist software managers in putting together the right team. These ideas serve as foundations for this research study.

Chapter 2 presents a general background on prior research in building teams, and other related research on personality, culture, knowledge, experience, skills, and other relevant psychology issues. Chapter 3 discusses the different roles of software engineers, describing the role of a project manager, a team leader, a requirements engineer, an architect, and a designer. Chapter 4 presents the Belbin Model, the Myers-Briggs Model and Keirsey Theories, as well as other team building models. Chapter 5 discusses cultural aspects, exploring the role of culture in team management, as well as other relevant information to cross-cultural communication, negotiation, decision making, and leadership. Chapter 6 involves the discussion of this research study and proposes a model that could work best for assembling teams for software projects based on previous arguments and studies. Chapter 7 includes the conclusion and future work.

Chapter 2

Background

2.1 Building Teams

The complexity of the problems that project managers face today requires multiple perspectives and varied expertise. This weighs heavy on project managers as they consider how the combination of people with respect to knowledge, skills, culture, language, and aspiration matches the outcomes that they are willing to produce. Managers must assign tasks to team members by assessing their capabilities in order to avoid confusion, conflicts, and disappointing results (Canning 2005). Team members need to have the capability to work together, take responsibility as individuals, work in functional groups, and join the project groups for integrating specific tasks into a larger system (Galton 2003). Key elements of building effective teams are:

1. **Style and Cohesion** - The cultural style and dynamics of a team are affected by team members' experiences, as well as by their beliefs. Therefore, a properly functioning and cohesive team would provide social and emotional support for all individuals in order to gain the required knowledge, improve their skills, and deal with unexpected issues as they arise (Galton 2003).
2. **Identification and Internalization** - A successful team consists of a stable and effective relationship between members in a team, and with other groups;, these relationships should be "positively constructive, mutually rewarding and mutually satisfying" (Galton 2003).
3. **Motivation and Improvement** - A positive motivation that propels a project forward to success would be generated from previous experiences, beliefs, and values of team members. Team members should have good communication skills to encourage active participation among the entire team. On the other hand, a team leader should have the ability to successfully combine different individual skills with various personalities to form a highly motivated team. Good team

dynamics and team spirit depend on the level of interaction between individual personalities within a group (Galton 2003).

4. **Models and concepts** - The Belbin model is one of the most recognized models for team building. It relies on a questionnaire that produces a useful and unique solution for assembling teams based on roles. Using this model, managers can decide how to allocate work for maximum motivation and productivity, as well as to assign tasks to individuals for the best fit (Galton 2003). The Belbin model and other team formation models, such as the Myers-Briggs model that relies on personality types, are discussed in more detail in Chapter 4.
5. **Team Building Factors** - There are many factors that affect the performance of a team such as team size; work environment; skills and styles of team leaders and facilitators; personality traits; beliefs and values; abilities; team dynamics and spirit; learning styles and coaching methods; individual and collective experiences; and roles and responsibilities (Galton 2003).

Parker (Parker 2000) states that "Not every group is a team, and not every team is effective." The best software team structure depends on the management style, the people who form the team with respect to their different level of skills, knowledge, and experience, as well as the complexity of the problem to be solved. Software teams should be structured with respect to:

1. The difficulty of the problem that will be solved,
2. The size of the resultant programs in lines of code or function points,
3. The time that the team will stay and work together,
4. The degree to which the problem could be modularized,
5. The required quality and reliability of the system to be built,
6. The inflexibility of the delivery date, and
7. The degree of sociability that is required for each project.

Software managers should be wary of not providing a clear definition of roles to all team members (Pressman 2005). On the other hand, Constantine (Constantine 1993) recommends four different paradigms that could be implemented in organizations for managing software teams:

1. A closed paradigm: Builds a team that follows “a traditional hierarchy of authority”. In this case, team members can work and produce software that is almost similar to previous developed products. However, individuals in this closed paradigm will not have the opportunity to be innovative.
2. A random paradigm: Structures a team loosely and counts on the level of creativity and innovation of the individuals. This paradigm works best for projects that require innovation or technological breakthrough.
3. An open paradigm: Organizes a team that works collaboratively, and communicates regularly, but is controlled by the rules and procedures of a closed paradigm as well as the flexibility of a random paradigm in order to allow team members to share their innovation. This paradigm is useful for solving complex software problems, but may not be efficient all the time.
4. A synchronous paradigm: Depends on the nature of the problem that needs to be solved; a team is organized in a way that enables team members to communicate and focus on pieces of the problem.

2.2 Psychology Background

Due to the complexity of human interactions many project team problems may not be so much technological, but sociological in nature. Effective teamwork by smart and hardworking people motivates team members, avoids conflicts, and saves time. “Team formation is chemistry due to a mix of competence, trust, mutual esteem, and well person sociology that provides perfect soil for the growth of jelled teams (DeMarco 1987).” Demarco (DeMarco 1987) presents the term “Jelled Team” as a group of members so strongly close to each other that the entire team is greater than the sum of its parts – synergistic. Individuals of jelled teams are more productive and motivated since they share a common goal and a unique culture.

Other psychologists have different definitions for the term "Team". For example, a team is defined as "a set of two or more individuals who are connected through interaction" (Biddle 1979). On the other hand, others defined a team as a group that satisfies the following criteria: have at least one task to perform, have different roles, interact within a social system, and work within a particular environment and boundaries (Hackman 1990). Katazenbach (Katzenbach 1993) describes a team as "a small number of people with complementary skills who are committed to a common purpose, performance, goals, and approach for which they hold themselves mutually accountable."

Researchers have found that psychological characteristics at the individual level are relevant to organizational outcomes, since personality and values have a substantial impact on team performance (Poling 2004). It is found that successful teams have diverse personalities in which human aspects of project development are more essential than technological difficulties. Furthermore, personality type analysis can aid managers in selecting the right combination of people to form an effective team (Gorla 2004). Myers (Myers. and Briggs 1987) studied the personality aspect and developed the Myers-Briggs Type Indicator (MBTI) that illustrates the impacts of personality characteristics on software teams. More information relevant to personality characteristics is discussed in the following section.

On the other hand, role theory is defined in various ways. Sarbin (Sarbin 1954) describes the role theory as "an interdisciplinary theory in that its variables are drawn from studies of culture, society, and personality." Biddle (Biddle 1979) defines the role theory as "behavior characteristics of one or more persons in a context." Further, Biddle defined the role functions as a "behavioral repertoire characteristics of a person or position." Roles can also be evaluated in terms of their consequences such as their "characteristic effects, or functions, within a societal system." Biddle supports using personality characteristics or types in order to investigate different perspectives of roles.

2.3 Personality

Software engineers may have the right technical skills to work on a software project, but do the team members have the right personality (Howard 2001)? The major cost item for software development projects is the poor staff selection, since people's personalities have a substantial impact on improving the chance of a project's success. People work in various ways, develop individual problem-solving habits, and have different understandings of the requirements of the tasks, dictated by their personality types. Furthermore, different mentalities analyze the problems in various manners (Howard 2001).

Howard categorizes seven different work personalities (Howard 2001):

1. Delivers: Those people focus on the activity and they are capable of producing a good product within the first trial. They work effectively in short-term tasks or rapid application development projects, as well as make good emergency technical or maintenance staff.
2. Prototypers: Those people work best on projects where the requirements are not clear. They prefer to plan, build, and see something working and then revise it over time.
3. Perfectors: Those people fit best on safety-critical applications where it is important to take details into consideration. They build module by module, piece by piece, and make sure that they are doing the right thing after completing each step, until the entire system is integrated successfully.
4. Producers: Those people prefer to work in their own way and use new methods, techniques, and standards to produce programs and systems that operate without asking too may questions or getting directions from project managers.
5. Fixers: Those people are familiar with the system they used to work on so they can make fast changes in hours or days that could take new developers weeks or months. They can meet customer requirements very quickly and do maintenance very efficiently.
6. Finishers: Those people make sure to complete a project on time since they just focus on finishing their tasks. They follow whatever rules, procedures, and behaviors are necessary to finish and deliver the final product on time.

7. Non-Finishers: Those people tend to put things behind and always look for excuses, such as saying that the rules and regulations are the main cause for not completing their tasks. They pretend that they are progressing but in reality they are not.

Software development projects should consist of a combination of people who are convergers as well as divergers in order to deliver the right system at the right time. For example, when a converger shows a sense of urgency, it may motivate a diverger to come up with a creative solution. The following table (table 2.1) shows the differences between convergers and divergers (Howard 2001).

Table 2.1: The differences between Convergers and Divergers

Convergers	- Have scientific and engineering backgrounds - Good at design analysis - Good at following work plans and procedures - Focus on problems and quickly classify them - Good at applying tools and techniques to problem-solving - Prefer to come up with solutions as soon as possible - Able to redefine problems to be fitted within their techniques but sometimes lead to a complex solution
Divergers	- Less technical background - Good at creative thinking - Good at discovering new situations - Less happy following the rules and procedures - Prefer an unstructured approach to analyze their problems - Leave issues and problems to grow - Prefer to spend more time formulating the problem and try to find better solutions

Other researchers have different categorizations for personality types such as Myers-Briggs personality types (Lawrence 1994; Layman 2006). Chapter 4 discusses the differences between introverts and extroverts, sensors and intuitors, thinkers and feelers, as well as judgers and perceivers.

Belbin (Belbin 1981, 1996) also produces four types of personality which are:

1. Stable extroverts: Those people are excellent in fulfilling duties, tasks, and promises and excel in their work, especially the jobs that require an individual to

cooperate and communicate effectively with others in order to gain valuable information. Additionally, they do a good job in personnel management.

2. Anxious extroverts: Those people are helpful when team members are supposed to work at very high speed and make progress under extreme pressure in order to get the job done as soon as possible.
3. Stable introverts: Those people prefer working on an environment that consists of high-quality relationships with a small number of individuals who need to work continuously with each other over a long period of time.
4. Anxious introverts: Those people are creative, persist in working alone in long term projects by following their own directions and procedures, and have the ability to handle critical tasks and duties.

Software teams often struggle with the differing personality characteristics of its individuals. More specifically, some individuals are extroverts and others are introverts. Some team members collect information intuitively and focus on facts more than on broad concepts. Others process such information linearly and gather all available details from that data. Some individuals have the ability to make decisions when they have a logical argument to depend on, while others prefer making decisions based on how they feel about a particular argument. Some practitioners prefer a detailed schedule with organized tasks to follow. Others prefer more open issues that are not suggested or planned. Some people do their tasks in advance in order to avoid the stress of reaching the deadlines, while others prefer to be in a rush and work at the last minute. This means that it is quite important to study the recognition of personality differences while structuring software teams in order to create effective ones (Pressman 2005).

2.4 Culture

Cultural is a key consideration for this research study, since project managers today interact with team members who have different cultural perspectives. To this end, project managers must understand patterns of beliefs, values, and behaviors shared by project team members in order to effectively plan, communicate and control the team, handle risks, and make the right decisions.

Culture is generally defined as “the arts, beliefs, customs, institutions, and all other products of human work and thought created by a people or group of people at a particular time.” Cultures are classified in various ways: some people compare Eastern to Western cultures while others study the cultural differences between countries, regions, states, cities, and towns. Other researchers break down culture into subcultures dealing with religious, racial, linguistics, and ethnic backgrounds (Sherriton 1997). The important issue is that project managers live and deal every day with different people from all over the world, which requires them to carefully understand how those people think and behave in order to effectively form and manage good teams.

Krishna et al. (Krishna 2004) investigate cross-cultural issues in North America, Western Europe, Japan, and India. They find that societies tend to have different ways of working and communicating. For example, Indian software organizations have discovered that they need to communicate with American people and Japanese in different manners, since Americans tend toward extensive written arguments and explicit documentation and prefer to contact other parties using informal email or telephone. However, the Japanese prefer verbal communication, more negotiated arguments, less words, and more formal use of email and telephone. Furthermore, they observe that Indian programmers would prefer not to criticize in face-to-face meetings; instead, they tend to send their comments and opinions in email messages outside of the meeting time. On the other hand, British managers feel frustrated if someone has an idea or a suggestion and did not propose it right away during the meeting time. They found that Germans do not mind staying later at work, while Japanese do and try to avoid working overtime as much as they can , especially when they work outside of their own country.

Setlock et al. (Setlock 2004) study the cultural effects and variations between three groups: American-American (AA), Chinese-Chinese (CC), and American-Chinese (AC), and conduct useful evaluations based on two decision-making tasks, one face-to-face, and the other using instant messaging. They notice differences in conversational efficiency, conversational content, interaction quality, and persuasion, but not in performance. More specifically, AA groups were the most effective in the communications medium. However, the CC and AC groups were least effective in face-to-face and instant messaging. The authors observe that CC pairs have problems using a second language to perform their tasks, and always try to build a deeper cognitive agreement. AA and CC pairs analyze the problems in different ways. "Eastern managers sought more understanding of implicit meanings and multiple cues in decision-making tasks, while Western counterparts relied on readily accepted sources of information with explicit meanings." It also observed that CC groups tend to show more respect, avoid overt disagreements, ask for opinions, and use more politeness expressions than AA or AC groups. In addition, CC pairs were more persuaded by one another while Americans tend to have, to some extent, a lower level of persuasion. On the other hand, the researchers do not recognize any cultural variations in task performance between the three groups.

Massey et al. (Massey 2001) discover that Korean people prefer to work and complete tasks collaboratively with team members while Americans and British are more comfortable with "loose ties" among team members and the division of work. Asians and Middle Easterners tend to use high context communication in their social interaction while Americans prefer using low context communication. The authors notice that Japanese always ask for details about plans and procedures and have a lower tolerance for uncertainty and vagueness, while Americans and British prefer fewer rules, less structure, and are more comfortable with vagueness.

Cultural aspects are very important to be taken into consideration for this research study, since project managers today deal with team members who have different cultural perspectives. As indicated earlier, project managers should understand patterns of beliefs, values, and behaviors shared by project team members in order to effectively plan, communicate and control the team, handle risks, and make the right decisions. Furthermore, project managers can have a better understanding of how each individual could feel, think, and behave or react to a specific situation, as well as how they tend to exchange information, share ideas, and make successful changes. Cultural aspects including roles, variables, values, and profiles as well as relevant information to cross-cultural communication, negotiation, decision-making, and leadership are discussed in chapter 5.

2.5 Knowledge, Skills, and Experience

Knowledge, skills, and experience are the traditional decision vectors for team member selection. When a project requires a technical lead, these are often the key decision criteria. However, these alone often result in mismatches with other key aspects of the project team dynamics as the team members interact to accomplish their tasks and ultimately deliver the software products. For example, if a team lead is a sound technical background but has a belief system that is at odds with the project manager, this conflict can drain much energy from the team as they struggle with each other for control. In this subsection, we examine these aspects individually to provide a sense of base criteria from which most team selections are made. Recognize that this research endeavors to extend these aspects with more social models to result in better selections for high performing teams.

2.5.1 Knowledge

Developing software requires both technical and functional domain knowledge; a software design team should have enough knowledge distributed between all team members in order to successfully complete building an application that matches the customer's requirements. In general, software team members do not have all the required knowledge for a particular project and should acquire that knowledge and additional information before starting work on the project. Individuals can find the required information from previous relevant documentation, formal training sessions, the results of trial-and-error behavior, or by gaining some knowledge from other team members. Design activities need to take place around the integration of the several knowledge domains. This integration aids team members in sharing their knowledge and in coming up with different models and potential solutions. These models develop over time during which individuals get the chance to learn from one another about the expected behavior of the application as well as the required computational structure in order to produce this behavior. Thus, team members should speak the same language in order to be able to facilitate communication and understanding (Walz 1993).

Walz et al. (Walz 1993) examines how group members acquire, share, and integrate project-related knowledge by increasing the amount of application domain knowledge among the whole software development staff. Moreover, assigning at least one knowledgeable member with deep application domain to a design team could significantly reduce the learning time and help resolve conflicts within a short period of time. They observed that to cover the different knowledge domains of a specific project, proper staffing is required. Also, knowledge acquisition should come at the first stage as the highest priority before dealing with engineering the requirements into an appropriate design.

2.5.2 Skills

To lead successful software engineering teams, it is important to understand their structures as well as the skills of the team members. Software teams should be an assembly of individuals that have a variety of skills such as communication skills, teamwork skills, interpersonal skills, strong work ethic, analytical skills, sharing and collaborating skills, adaptability and flexibility, honesty and integrity, computer skills, and organizational skills. Team members should have the required skills to communicate in order to exchange information, practices, and concepts that are related to a particular software project, as well as explain ideas to those who are not familiar with a specific concept. Furthermore, individuals should be able to cooperate and participate in order to solve complex problems and develop the desired software. On the other hand, team members should demonstrate their willingness to improve and to care about their teamwork. Moreover, team members should be willing to learn and take more steps to search for material related to their projects. One of the most important characteristics that makes a project successful is to have a combination of people who are open minded and willing to listen to others in order to learn new things, and people who have good visions for the future (McKinney 2005).

Javed el al. (Javed 2004) show that effective team communication plays a vital role in increasing the performance of employees, as well as enhancing the software development productivity. Prior studies show that organizational communication is essential since it focuses on improving the interpersonal skills of project managers, specifically speaking and writing skills. However, inter-organizational communication has become an important issue today in software team management, since it concentrates on improving the interpersonal skills of team members. The studies also show that an employee who works for a long time in an organization is more satisfied with communication than an employee who has just joined the organization. This research also illustrates that good personal behavior with project managers has a positive impact on improving work efficiency. A good work environment has a substantial effect on enhancing communication among employees as well as increasing their work competence. Therefore, organizations should provide workspaces that are noiseless, spacious, private, and non-interruptive. Team members who are willing to support each other in work-related problems are more efficient than members who are withdrawn and unhelpful. Moreover, teams' members feel more comfortable with project managers who let them communicate and express their ideas about work-related matters and suggest solutions. Thus, team members produce work more effectively when they get feedback about their performance such as quality of work, top management evaluation, and personal appraisals.

Beaver and Schiavone (Beaver 2006) study the effect of the skills and experience of the software development team members on the quality of the final software product. They found that the skills of software team members have a significant influence on both the design and implementation. Furthermore, the increasing presence of less skilled individuals in each phase of the development life cycle decreases the chance of getting high quality software. Moreover, people who have sufficient skills for requirements engineering and design produce high quality design modules and source code units, as well as ensure the completeness and correctness of the entire software product.

Cushing et al. (Cushing 2003) identify three categories of skills related to software development projects: tactical project skills, intra-and inter-personal skills, and management skills (See Table 2.2). In addition to these skills, developing self-awareness assists team members to evaluate their own initiative, collaboration, and accountability levels and to monitor their progress over time.

Table 2.2: Team skills needed for software development

Skills	Benefits
Tactical skills	- Organize work - Determine tasks - Assign roles - Make decisions - Create Plans - Solve problems
Intra-and inter-personal skills **Communication skills** **Intermediate communication skills** **Advanced communication skills**	- Complete the work and accomplish the project goals - Ask questions - Listen and explain - Give and receive feedback - Resolve conflicts - Give presentations
Management skills	- Lead a team - Conduct meetings - Coordinate activates

As a result of Cushing's (Cushing 2003) study on how to create an effective team based on improving individuals' skills, the authors mapped team skills onto the software development life cycles in which the requirements engineer, system analyst, and system designer should have enough tactical skills in order to make decisions. On the other hand, the system analyst, system designer, software tester, and software implementer should have the required skills to solve problems. Every team skill is required at each stage of software development; some of them are more important than others in some phases. It is concluded that the system analyst and system designer should have most of the skills to perform their jobs.

Another study (Bailey 2001) identifies the skills and abilities that are required for computer programmers and software developers. The findings of this research demonstrate that developers need a mix of three types of skills: technical skills, soft skills and business concepts as shown in Table 2.3, where technical and soft skills are most important to IT professionals.

Table 2.3: Three types of skills for IT professionals

Skills Categories	Description
Technical skills	- Ability to read, understand, and modify programs written by others as well as code programs and debug software - Knowledge of structured programming fundamentals, multiple programming languages, design methodologies, design specifications, project management, database normalization, technological trends, client/server systems, software development tools, object oriented concepts, operating systems, SQL, GUI design, DBMS, C++, JAVA, UNIX, network fundamentals, Oracle, Visual basic, HTML, MAC OS, ergonomic interfaces, and web authoring tools - Ability to implement programs - Ability to read design specifications for conversion to code - Ability to read and write technical documentation - Ability to design software programs and user friendly applications - Ability to estimate project time, and derive project plans - Ability to install software and use CASE tools - Ability to research language syntax - Ability to troubleshoot hardware
Soft skills	- Ability to listen, solve and analyze problems, create decision tress, multitask, and understand business culture - Teamwork skills, time management skills, verbal communication skills, inter-team communication, organizational skills, interpersonal skills, stress management skills, general writing skills, technical writing, leadership, and presentation skills - Ability to visualize and conceptualize, apply knowledge, give and receive constructive criticism, as well as understand different cultures - Adaptability to new technologies and languages
Business concepts	- Project management - Investigative skills, idea initiation skills, interviewing skills, mediation skills, role playing skills, and basic accounting skills - Ability to read and understand budget as well as documentation - Knowledge of marketing concepts

2.5.3 Experience

Most people acquire their experiences over time from either learning and applying, or working in organizations. Highly trained and skilled people often produce high quality work. However, inexperienced software developers have a significant adverse effect on the quality of the software products (Beaver 2006). Therefore, software teams are encouraged to have some experts, since they present the source that provides critical information and knowledge to all team members. This helps individuals trust themselves while making their decisions, which speeds up the process of completing the various tasks.

Faraj and Sproull (Faraj 2000) investigated the importance of expertise coordination and collaboration among 69 software teams. They discovered that there is a strong relationship between expertise coordination and software team performance that remains significant over team input characteristics, presence of expertise, and administrative coordination. Generally, software teams are structured based on the project requirements and it is rare for an entire team to move together from one project to another. Therefore, teams never develop a history that includes the work of a team over multiple projects. This means that at least one expert needs to be involved in every software team in order to ensure high quality work by conveying specialized skills and knowledge to all team members.

Research on software teams has found that the performance of a team is connected with the effectiveness of teamwork coordination (Kraut 1995; Faraj 2000). Coordinating expertise in software development teams allows team members to manage conflicts, avoid difficulties and share knowledge. Team performance is not just a function of having the right expertise on the software team. Expertise should be coordinated among individuals. Furthermore, expertise management includes the evolution of socially shared cognitive processes to satisfy the demands of task based skills and knowledge dependencies. In general, every software team should involve some experts in order to help other team members understand their tasks, assignments, and roles, and to provide different kinds of solutions and ideas for solving critical problems.

Another study (Chung 1994) demonstrates that the professional experience of members of software development teams moderates the relationship between participation and performance. The participation of expert individuals has a substantial effect on team performance. As a result of this study, Chung found that participation of experts within software development teams allows team members to effectively develop plans and procedures, as well as to translate broad goals into operational plans. Additionally, team members did a good job of managing their activities and were able to define their tasks successfully. Individuals were able to monitor the flow of the work among team members, define priorities, and specify the most important aspects of the work. Experts let other team members learn how to listen to the users and understand their needs. In addition, they taught them how to communicate with one another during requirements analysis and work as a team. On the other hand, managers also reaped the benefits of consulting some experts for their ideas and advice.

Chapter 3

Software Engineering Roles

Software engineering as a discipline consists of many roles and responsibilities from a project team perspective. In this section, we examine the role of a software project manager describing the characteristics of effective project managers, and presenting the capability maturity model for project management. We also examine the role of a software team leader describing the characteristics and key roles of an innovative team leader. Moreover, we examine the roles and tasks of a software requirements engineer, architect, and designer.

3.1 Project Manager

While software project management differs from organization to organization, it largely consists of the planning, monitoring, and control of the individuals, process, and events that arise as software evolves from a preliminary concept to an operational deployment. At the beginning of each software project, the project manager often develops a business case with or for executive management, identifies the product requirements, acquires funding and management sponsorship, forms the team, develops estimates and a general project plan, and obtains other required resources (Wiegers 2005). Software managers are responsible to deliver a high quality software product on time and within budget. Therefore, project managers are responsible to recruit people and structure software teams, as well as choose an appropriate process for both of the team members and the product (Pressman 2005). Software project managers must allocate team members to roles, responsibilities, and tasks that minimize the required time staff spend on working among themselves to maximize their efforts and activities. Therefore, managers should assess the personalities of the team members and try to make the best use of them within the project's constraints by fitting them into their best positions (Howard 2001).

Project managers must also make plans by estimating effort and calendar time that are required to accomplish a series of work tasks for a particular project. The work tasks that should be accomplished by software managers are: defining work products, creating quality assurance checkpoints, and determining mechanisms to monitor and control the entire project plan. The plan involves definitions of the process and tasks to be conducted; the individuals who work on the project; and the mechanism for mitigating risks, controlling unexpected changes, and assessing quality. In addition, software managers should keep in mind that software development depends on human activities; successful managers encourage comprehensive stakeholder communication early in the project evolution, handle the risks and propose proper solutions (Pressman 2005).

More specifically, senior managers coordinate the interface between both the business specialists and software professionals. They determine the business issues that often have a significant effect on the project outcome. Project technical managers focus on planning, motivate, organize, and monitor the team members who work on producing the software products (Pressman 2005).

3.1.1 Characteristics of Effective Project Managers

Characterizing software project managers is addressed widely both in industry and research publications. And as such, there are a number of varying opinions and camps that describe the capabilities of a software project manager. Edgemon (Edgemon 1995) suggests that effective project managers should emphasize four characteristics:

1. Problem solving: The ability to determine the most relevant technical and organizational tasks to a particular project, discuss different possible solutions with team members, be flexible enough to listen to other solutions, make use of knowledge gained from previous projects, be flexible enough to make changes, and build an appropriate solution.
2. Managerial identity: Having the confidence to control people and let them follow rules and procedures, as well as make critical actions when necessary.
3. Achievement: Giving rewards for the accomplishments and productivity of project teams.

4. Influence and team building: The ability to know people, to understand verbal and nonverbal signals, responds effectively to their concerns, and remain under control in highly stressful circumstances.

3.1.2 The Capability Maturity Model – Project Management

The capability maturity model (Wiegers 2005) suggests that software project managers should manage and control people, communication, commitments, requirements, resources, change, risks, opportunities, expectations, technology, and suppliers, as well as deal with conflicts. Furthermore, software managers should perform the following tasks:

1. Define Project Success Criteria

Project success is defined as satisfying the set of requirements and constraints based on stakeholders' expectations. Thus, a software project manager is responsible for defining success criteria that keep stakeholders focused on shared purposes, and establishes plans for assessing progress in order to meet the requirements. Defining the success criteria at the early stages of any software project also helps project managers avoid a disappointing business outcome. Additionally, it assists project managers who work on subprojects in integrating the small units into the large system.

Business objectives typically include the expected outcome, the time frame for accomplishing the objective, and the evaluation method for obtaining successful achievement. In other words, it consists of what the product should be doing, including important or distinguishing functions. In addition, business objectives involve economic constraints, including development costs, materials costs, and estimated time to market, as well as the product's operational and temporal context, such as technologies that are required to assure compatibility in the target environment, backward compatibility, and future extensibility for software enhancement. Therefore, a project manager should write the business objectives that are specific, measurable, possible, relevant, and time-based.

Moreover, a project manager should be flexible enough to modify the business objectives in case of discovering unexpected situations: for example, if a project manager figures out that the cost of materials might be higher than the estimated or the final product would exceed the total cost and will not meet the reusability goal.

A software project manager should identify internal stakeholders: the team leader and the team members, including analysts, developers, requirement engineer, architect, designer, testers, and technical writers. Also, a project manager should determine external stakeholders: government agencies, subcontractors, customers, business partners, and materials, information, and service suppliers. Furthermore, a project manager should specify stakeholders' interests, which involve financial benefits, exact time to market, functionality, performance issues, usability, reliability, and other quality aspects. Moreover, the project manager should identify a decision making process and select a proper decision rule which consists of voting, reaching unanimous agreement, achieving consensus, and asking a team member to make a decision based on team opinions.

A project manager should also identify project constraints by determining boundaries and limitations within which the software team must perform. Furthermore, the project manager should have the ability to make tradeoffs along these dimensions: features, quality, staff, cost, schedule, and risk, and be able to adjust or prioritize them according to the nature of the projects. On the other hand, a project manager should monitor the progress of the software product, checking both the functional and nonfunctional requirements, and make sure that performance, reliability, and throughput goals meet the business objectives before releasing the final version.

2. Define Product Vision and Project Scope

A software project manager is responsible for describing a solution that meets the project business objectives and identifying the subset of the last vision that the current project will address with respect to the limitations forced by project constraints.

3. Define Product Release Criteria

A software manager should decide what criteria will indicate whether the final product is ready for release or not. The criteria should be realistic, objectively measurable, documented, and satisfy customers' requirements as well as assure the product's quality (Weigers 2003).

4. Negotiate Achievable Commitments

A software project manager should be realistic and not make a commitment that could be impossible to keep. Instead, the software manager should make achievable promises and successful negotiations about project goals, objective criteria, interests, difficulties, functionalities, schedule, staff, budget, and deadlines when arguing with customers or team members. The project manager should also be able to distinguish between the problem and the people (Weigers 2003).

5. Study Previous Lessons Learned

A software project manager should review previous projects and get concrete ideas in order to make persuasive arguments and manage project risks.

6. Conduct Project Retrospectives

A software project manager should be able to successfully perform post-project reviews for all team members, capturing lessons learned from prior projects or phases in order to improve their future performance. This would also help the project manager and team members repeat their successful approaches, share experiences, and avoid facing similar problems in the future (Weigers 2003).

3.2 Team Leader

Team leaders are responsible for organizing project teams in appropriate ways that maximize each team member's skills, abilities, knowledge, and experiences. Weinberg (Weinberg 1986) presents a MOI (Motivation, Organization, and Innovation) model of leadership:

1. Motivation: The ability to motivate and encourage technical team members to produce to their best capability, complete tasks, and achieve goals.
2. Organization: The ability to understand and use the existing processes or create new ones that would translate the initial concept of software building into a final product.
3. Innovation or Ideas: The ability to encourage team members to be creative or innovative even if they are working within constraints that are determined for a specific application or product.

The aim of any team leader is to accomplish something quite different, which means not only doing things differently but also leading in a different way. An innovative leader is the one who has a control style which allows team members to feel free to jump onto a problem once it appears, and to approach unique solutions in different ways. Moreover, a team leader should push the entire team towards high performance objectives, and facilitate collaboration and communication. Furthermore, a team leader should not interfere with detailed team activities since micromanaging a project can lead it to fail. A team leader should also give the opportunity to all team members to improve their arguments and judgments in decision making and action taking (Harris 2003).

3.2.1 Characteristics of a Team Leader

This subsection describes the different characteristics of a team leader. Harris (Harris 2003) suggests seven characteristics of a team leader:

1. Holistic: Those leaders have the ability to communicate and exchange information with all team members who have good domain knowledge, unique experience, different ways of thinking, and different mentalities and ideas. Furthermore, those team leaders are able to receive and integrate business theory and practice based on their highly developed knowledge and experiences. They are familiar with the nature of complexity of various challenges. More specifically, they recognize both the internal and external uncertainties and are capable of building teams that provide unique system solutions.
2. Empathic: Those leaders have the ability to understand the emotional needs of team members as well as their personal goals and aspirations. They know how to show their sensitivity and deal successfully with different situations that could affect members' emotions and cause the whole team to fail, or at least get behind schedule.
3. Patient: Those patient leaders can easily improve and motivate a team because they often allow people to work on their tasks comfortably, without any stress, in order to get accurate results at the end.
4. Supportive: Those leaders have a supportive nature that allows them to cover or back up their team members when necessary. In other words, they give useful directions to individuals who lose their path in order to keep their projects on track and making progress,, and build good reputations that other team members don't forget. .
5. Virtuous: Those leaders have the quality of being honest and moral, and have good ethical management principles. This helps team members to be honest, and to trust and respect their leaders.
6. Considerate: Those team leaders care about the well being of their team members, showing kindness and appreciation.

7. Dogged: Those team leaders insist upon solving or completing something effectively and getting the job done. They motivate people to accomplish tasks and produce perfect outcomes.

3.2.2 Key Roles of an Innovative Team Leader

Harris (Harris 2003) also proposes seven key roles of an innovative team leader:

1. Build commitment: Those team leaders have enough skills that enable them to build commitments across the entire team with a sense of destiny and a sense of purpose. They have the ability to think in the longer term, look at the future, and imagine what is necessary.
2. Constant feedback: Those leaders provide valuable feedback regularly to all team members to learn, fix, adjust, and improve as the team progresses. They discuss their comments collaboratively to allow individuals get the benefit of that feedback.
3. Build confidence: Those team leaders are capable of building confidence for all team members and the whole team, motivating and showing them the early wins and reminding them of their achievements.
4. Facilitate functional skills building: Those leaders have the ability to teach individuals how to excel, building upon their skills and knowledge. Furthermore, they are able to convey their experiences to motivated team members and let them use those experiences effectively to do their tasks.
5. Set high expectations: Those team leaders always ask individuals to work hard and produce products that satisfy a very high level of expectations in order to increase their performance and obtain high quality outcomes.
6. Focus on high performance goals and metrics: Those leaders facilitate project targets but at the same time encourage team members, asking them to specify their own goals. Thus, individuals will set high goals, and will never hesitate to achieve them. Additionally, those leaders evaluate the project team's performance against the project objectives in order to ensure that everything is under control.

7. Facilitating team building: Those team leaders concentrate on project aspects and team building issues simultaneously in order to construct high performance teams which can produce successful innovation. Moreover, those leaders should have the ability to select their staff based on their personality, knowledge, experience, skills, and background.

3.3 Software Requirements Engineer

Requirements engineering is the most important phase in software development life cycles since it identifies both functional and non-functional requirements, and provides both the architect and the software designer a complete and accurate specification (Pohl 1995). Requirements engineering consists of a set of tasks that aid software engineers to recognize the problem they will be solving, understand the business impact of the software, realize customer needs, and determine how end users will interact with the final product. Software requirements engineers communicate with the customer and other stakeholders, such as managers and end users, in order to understand the product scope and requirements. The requirement engineers are responsible for focusing on both customer needs and software product quality. Understanding the requirements of a problem is difficult, since end users do not have a good awareness of what functions and features their final system should provide. Also, requirements always change throughout the project, which makes it difficult to reengineer. Generally, Software engineers struggle with customers when trying to elicit their needs, and have trouble interpreting the information they receive. Thus, software requirements engineers should understand the customer needs in order to design and build the right computer program that solves their problems (Pressman 2005).

Requirements engineering is “a software engineering action that begins during the communication activity and continuous into the modeling activity (Pressman 2005).” It constructs a bridge to design and implementation once business needs are specified, user scenarios are explained, functions and fearers are outlined, and project constraints are determined (Pressman 2005).

3.3.1 Requirements Engineering Tasks

Requirements engineering supplies the suitable method for understanding customer wants, recognizing needs, evaluating feasibility, negotiating a logical solution, suggesting unambiguous solutions, verifying and validating the specification, and managing the requirements as they are moved into an operational system (Thayer 1997). The requirements engineering tasks are (Pressman 2005):

1. Inception: The requirements engineer defines the scope and nature of the problem that needs to be solved. The requirements engineer starts asking the customer a number of questions in order to establish a basic understanding of the problem, people who will use the system, and the nature of the desired solution, and to develop an effective communication channel between both the customer and the developer.
2. Elicitation: This task helps the customer defines what he wants. However, understanding clearly what the customer needs is hard for many reasons:
    a. Problem of scope: Customers may identify unnecessary technical details that could confuse the developers rather than making the overall objectives clear.
    b. Problems of understanding: Customers do not know exactly what they want; do not have enough knowledge about the domain; have weak understanding of the facilities and limitations of their computing environment; fail to include important information that they believe to be obvious; do not have good communication skills to explain their requirements; ask for requirements that are not testable or unclear or lead to confusion; and determine requirements that conflict with the needs of other customers.
    c. Problem of volatility: The degree to which requirements change over time.
3. Elaboration: The requirements engineer is responsible for building up a refined technical model of software functions, features, and limitations based on the information that is gathered from the customer during inception and elicitation.

The requirements engineer depends strongly on understanding the user scenarios that show how the end users will interact with the final system, identify classes and services, and produce UML diagrams. As a result of this phase, an analysis model is developed which outlines informational, fictional, and behavioral domain of the problem.

4. Negotiation: The requirements engineer should negotiate with customers regarding limited business issues and conflicting requirements, asking all stakeholders to rank and prioritize requirements, and then discuss conflicts in precedence. Furthermore, the requirements engineer should specify and analyze risks associated with each requirement and do a rough estimation for development efforts in order to evaluate the effect of each requirement on the project budget and delivery time. As a result of this phase, the requirements are filtered, reduced, combined, and modified in order to get the final pure version of the requirements.
5. Specification: The requirements engineer is responsible for documenting the set of requirements using graphical models, formal mathematical models, a collection of scenarios, a prototype, or any mixture of these. The main purpose of documenting these requirements is to help developers have a better understanding of the function and performance of computer-based systems as well as the constraints.
6. Validation: Requirements validation examines the specification accurately, developing a set of checklist questions in order to ensure quality and verify that all software requirements are clear, all errors are detected and eliminated, and make sure that work products conform to the project, product, and process standards. A formal technical review is conducted by a special team that includes customers, end users, and other stakeholders, who validate requirements and look for defects, ambiguous terms, missing information, and conflicting and complicated requirements.

3.4 Software Architect

Architectural design represents the data structure and software components that are required to develop a computer-based system. It is a blueprint that provides a big picture of the system that consists of the architectural styles, the structure and properties of the components, and the interrelationships between the different systems' components that show how the components fit together. The architectural design is very important for building large and complex systems. Thus, the architect is responsible for choosing a proper architectural style for the requirements that are derived from the requirements engineering analysis. More specifically, the architect starts with data design and then advances to the derivation of at least one representation of the architectural structure of the whole system. Furthermore, the architect analyses alternative architectural styles or patterns to generate the structure that best satisfies the customer requirements and the quality attributes. Next, the architect elaborates the design using architectural design methods in order to get more useful details regarding the entire architecture. In addition, the work products that are developed in the architectural design phase are reviewed after each step by the architect to ensure the correctness, clarity, completeness, and consistency of the requirements (Pressman 2005).

The architect should have the ability to design an unambiguous blueprint that helps software engineers analyze the effectiveness of the design with respect to its requirements, present architectural alternatives if changes are required, and mitigate risks. Moreover, the architect should represent the architecture design of a system in a way that enables team members and stakeholders to understand in order to effectively communicate. The architect also should highlight early design decisions that will have a significant effect on the next phases as well as the success of the system (Pressman 2005).

3.4.1 Architectural Design Tasks (Pressman 2005)

1. Data design at the architectural level and component level: The architect is responsible for translating data objects into data structures at the software

component level, and a database architecture at the application level if needed. The architect also should apply the same systematic analysis principles for function and behavior into data; specify all data structures and operations; create a mechanism for defining the content of each data object and the operations associated with it; and establish a library of useful data structures and operations. Moreover, the architect should make sure that the software design and programming language support the specification of abstract data types, as well as ensure that the data structure representation is only known to modules that make use of the data.

2. Representing the system in context: The architect should use the system context diagram, which presents the user interface and the flow of information into and out of the system, in order to create a model that shows how the software interacts with external entities to its limitations such as devices, people, and other systems. Also, the architect should determine the structure of the system by defining and refining software components that apply to the architecture.
3. Defining archetypes: The architect should specify the abstract building blocks of the architectural design in order to design the system. Thus, the architect should define the following archetypes: node, detector, indicator, and controller.
4. Refining the architecture into components: The architect is responsible for refining the architecture into components that are derived from the application domain, the infrastructure domain, and the interface domain.
5. Describing instantiations of the system: The architect in this step develops a concrete instantiations of the architecture, mapping the architecture to a particular problem with suitable structure and components.
6. Evaluating alternative architectural designs: The architect is responsible for collecting scenarios, eliciting requirements, constraints, and environment descriptions, and describing the architectural styles and patterns that will addresses both the scenarios and requirements. Furthermore, the architect should evaluate quality attributes: reliability, performance, security, maintainability, testability, flexibility, reusability, interoperability, and portability. In addition, the architect should specify the sensitivity of quality attributes to different

architectural attributes for a particular architectural style, as well as critique candidate architectures using the sensitivity analysis.

7. Refining the architectural design: The architect should also refine the architectural design during early stages in order to ensure developing a blueprint that satisfies all functional and performance requirements based on design assessments.

On the other hand, Garlan (Garlan 1994) states that the software architect should participate and share knowledge throughout all the activities in the software development life cycle. The software architect also should recognize the role of software architecture in requirements engineering in order to understand the dynamic relationship between problem identification and solution description. Furthermore, the software architect and requirements engineer need to collaborate and work closely with each other to make sure that the requirements are consistent, correct, complete, clear, valid, and described in a proper level of details, as well as to ensure that the subsequent designs appropriately explain the requirements and constraints.

3.5 Software Designer

Software design consists of a set of standards, rules, procedures, concepts, and practices that help engineers and developers deliver high quality software products. It is one of the most important phases in software development life cycles during which, customer requirements, business needs, and technical aspects are integrated to build a system. The design model provides detailed information about data structures, interfaces, and components. Therefore, the software designer is responsible for modeling the system that is about to be built, as well as for evaluating and validating the designed model to ensure its quality (functionality, usability, reliability, supportability, extensibility, adaptability, serviceability, maintainability, testability, compatibility, configurability, and performance) before moving forward and starting to generate the software code. More specifically, the designer should model the system based on the architecture blueprint, components, and the interfaces that link the software to end users, systems, and devices. Following that, the designer assesses the model by detecting errors, testing alternatives, and enhancing the model, as well as making sure that the model could be deployed successfully within the project constraints, schedule, and budget. In general, software designers should have enough skills to produce a good model that supports the upcoming activities and provides an accurate translation of the customer’s requirements into a finished software product that is both stable and operates on real machines (Pressman 2005).

3.5.1 Design Tasks

This subsection describes the major tasks of a software designer. The software design engineer should perform the following tasks (Pressman 2005):

1. Examine the information domain model and design suitable data structures for data objects and their attributes.
2. Select an architectural style and pattern from the analysis model which best fits the software.
3. Divide the analysis model into design subsystems and assign them within the architecture, make sure that every subsystem is functionally cohesive, design interfaces for those subsystems, and distribute functions or classes to all subsystems.
4. Develop a set of design classes or components, translate every analysis class description into a design class, review each design class with respect to their criteria, specify methods and messages for each design class, assess and choose design patterns for a design class a subsystem, and regularly check design classes to ensure their correctness.
5. Design interfaces for external systems or devices if needed; those interfaces should be designed in a way that reduces the complexity of relations between components and the external environment.
6. Design user interfaces, check results of task analysis, identify an action sequence according to user scenarios, establish a behavioral model of the interface, determine interface objects and control procedures, and check the interface design when needed.

7. Perform component level design, identify all algorithms at a somewhat low level of detail, refine the interface of every component, determine component level data structures, and review all components and fix errors.
8. Build a deployment model and produce a design specification document that is readable and understandable for people who generate code and other individuals who test the system.

Chapter 4

Team Configuration Models

At the heart of this research project is the hypothesis that given existing team configuration and personality models, a software team configuration model can be derived that will accommodate roles and personalities, and introduce the notion of culture into the factors going into software team member selection. Hence, in this section we examine the Belbin roles and discuss some previous studies that applied Belbin role theory on software teams.

4.1 Belbin Model

R. Meredith Belbin's 1981 book (Belbin 1981, 1996) on management teams describes why they succeed or fail, defines eight roles based on personality types, and describes the different interactions that occur between the various roles while managing a team in order to help project managers appropriately build effective teams. The main reason for publishing this book was to provide project management the tools to assemble successful teams, since poor team building and poor interpersonal skills often result in week management. Belbin started analyzing and comparing the personality characteristics and critical thinking capabilities of individuals in order to determine the team roles (Jones 2000). Therefore, Belbin classifies the types of roles as either functional or team roles. The functional role is the role when the job requires a team member to work within a team based on member's technical skills and professional knowledge. However, the team role is the contribution that an individual makes to facilitate the progress of the whole team according to individual's behavior and the level of interaction (Gifford 2003).

Several questionnaires are used in this model to generate valuable and unique behavioral based evaluations of team members, teams and jobs. Furthermore, Belbin's role theory is helpful both for structuring teams from scratch for new projects, and for building existing teams. The benefits of determining an individual's role preferences are the ability to plan how to assign tasks for maximum motivation and productivity, and the ability to select people to fill gaps or allocate secondary roles for the best fit (Galton 2003).

Belbin observes that what “is needed is not well-balanced individuals but individuals who balance well with one another.” For example, Belbin finds that the majority of computer science and engineering professionals are high critical thinkers, and teams that consist of more or better high critical thinkers are often the least successful. This means that a good balance of team members has a substantial effect on the entire team’s performance. According to Belbin, “The success of a team depends on their team members’ roles’ interlocking pattern and how well they are discharged.” Generally, Belbin discovers by identifying the roles of a team, management would effectively apply individuals to the best functional roles (Belbin 1981, 1996). Belbin also suggests that personality and mental abilities of team members are two important factors that project managers can count on in order to assign appropriate team members to the different roles (Jones 2000).

The Belbin model helps project managers create profiles for all team members in order to get remarkable insight into individuals and team operating mechanisms. Thus, managers can have better understanding of a team’s and its individuals’ behavior by getting a profile of both team members and the whole team. The Belbin model also aids project managers in building and developing powerful teams with a balance of roles and skills by realizing individual strengths and weaknesses (Miller 2002). Furthermore, the model helps project managers arrange individuals’ technical resources to best advantage, especially when team members have balanced roles that ensure effective team work (Jones 2000).

4.1.1 Belbin Roles

Belbin characterizes the following roles that aid project managers in properly assembling effective teams (Belbin 1981, 1996).

4.1.1.1 Chairman (CH)

This team role identifies the controlling mechanism that leads a particular team to make progress towards achieving both the group objectives and organization goals. The chairman does this by being able to make the best use of team resources, being able to figure out where the team's strengths and weakness are, and to make sure that the best use of each individual's potential is done successfully. Generally, a chairman is a calm, self-confident person who controls the team as a team leader and has the ability to listen to other team members very well in order to get their ideas and opinions.

4.1.1.2 Shaper (SH)

This team role determines the controlling mechanism that will effectively allow team members to focus their attention on the core of objectives and priorities, and to enforce some shaper or pattern on group discussion and on the result of group activities. Generally, a shaper is also a team leader but has a managerial style and personality type that are different than that of a chairman. A shaper encourages individuals to work to their best abilities and try to find all possible solutions to the different problems. Shapers are extroverted, nervous, competitive, and argumentative.

4.1.1.3 Company Worker (CW)

This team role identifies the process by which concepts and plans are turned into practical working procedure and rules; approved plans are produced in a systematic and well-organized way. Company workers are conservative, dutiful, predictable, good organizers, hard workers, and self-disciplined. On the other hand, they are inflexible and more resistant to unproven suggestions, views and approaches.

4.1.1.4 Completer-Finisher (CF)

This team role determines the process by which team members focus on avoiding errors and inaccuracy of both commission and omission by effectively searching for work issues that need a lot of attention, as well as maintaining a sense of necessity within the whole team to meet deadlines and goals. Generally, completer-finishers concentrate on delivering products on time and within budget. Furthermore, those people are painstaking, orderly, conscientious, anxious, and illustrate a very strong sense of perfectionism.

4.1.1.5 Monitor-Evaluator (ME)

This team role identifies the process by which problems are analyzed and evaluates suggestions, methods, approaches, alternatives, and opinions to ensure that the entire team makes balanced decisions. Generally, monitor-evaluators are unemotional, dry, over-critical, hard-headed, and motivational to other team members. They always negotiate successfully with a Plant who comes up with innovative ideas, in order to make correct decisions and choices.

4.1.1.6 Plant (PL)

This team role identifies the process by which new methods, approaches, and ideas are developed, with particular attention to the most important aspects. The plant searches for possible breaks in the approach to the problems with which the group is confronted. Generally, the plant is introverted, unorthodox, imaginative, and intelligent. However, they tend to ignore details and protocols.

4.1.1.7 Resource Investigator (RI)

This team role identifies the process that leads to discovering and reporting ideas, developments, and resources outside the group, by developing external contacts that could be helpful to the team; this team member also carries out any following negotiations. Generally, the resource investigator is similar to the plant since both of them are creative and innovative. However, the resource investigators get their innovations from external sources because of their extroverted nature. They know how to communicate effectively with people and to get what they need from them, and have the ability to discover new things. However, they lose their interest in problems or circumstances when the novelty of it has disappeared.

4.1.1.8 Team Worker (TW)

This team role determines the process that supports individuals in their strengths and weakness, improves communications between team members, and helps the growth and development of team sprit. Generally, the team worker makes sure that the team works together toward meeting their goals and objectives. The team worker also facilitates or argues within the team to understand their needs. The team worker tends to be very social, mild, and sensitive; they are capable of responding, communicating, and dealing with individuals in different circumstances.

4.1.2 The Belbin Self-Perception Inventory (SPI)

It is important to consider the different roles discussed in the previous section for any project. However, how can these roles be evaluated for an individual? To address this question, Belbin designed a questionnaire that is used to collect data from team members in order to test the behavior of the whole team. This measurement provides indicators that help project managers make the best fit of an individual into each role. Stevens states that "One aspect of the test that affects how teams are set up concerns the fact that the numbers produced by the test are relative, not absolute." In general, the evaluation consists of various sections in which, for every single section, a team member allocates ten points among eight statements based on their feelings toward each statement. The Belbin Self-Perception inventory was created at the Administrative Staff College, Henley, by the Industrial Training research Unit from Cambridge (Stevens 1998). For more information related to the SPI, please look at (Belbin 1981, 1996).

4.1.3 Previous Studies

Many people have been interested in applying Belbin theory to software engineering projects. For example, Thomsett (Thomsett 1990) in 1990 implemented Belbin's role model on software development projects. He performed a qualitative analysis by which he discovered that there is some sort of similarity between the functional roles in the structured open team and some of the roles from Belbin theory. These similarities did not occur by chance, since the team role is considered as a completion of the functional role. This means that any kind of similarity between the functional role and team role would suggest a recommendation of a team role for a particular functional role. After two years, Yourdon (Yourdon 1992) supported the idea of using Belbin's role theory in software development teams in order to build effective teamwork. Generally, most prior studies showed enough evidence toward building effective software teams for real world development projects, especially in terms of programming contest scenarios, but the results demonstrated only trivial achievements (Gifford 2003).

After six years, Stevens and Henry (Henry 1998) started to investigate the quantitative relationship between Belbin's model and software development teams, but they were focusing their attention on building software teams based on simple class projects. These projects were completed in less than 90 minutes, located in one place, and allowed students to share a single machine. Thus, the overall results indicate that there is a quantitative relationship between Belbin's model and software development teams, but this does not completely reflect real world software projects.

4.1.3.1 A Study that Applies Belbin Leadership Role on Software Teams (Gifford 2003)

This section presents a study that has applied Belbin leadership roles to software development projects by setting up some sort of a real world experiment. This research shows that implementing Belbin's roles in software teams has a significant effect on improving team productivity, efficiency, and success in delivering high quality software on time and within the estimated cost (Gifford 2003).

4.1.3.1.1 Participants, Team Arrangements, and Constraints

108 Students from different sections of the "Introduction to Unix" class volunteered to participate and work seriously on this research as a part of their class project. All the students who participated in this research study were active, successfully submitted their assignments, and regularly attended classes. On the other hand, students who had major difficulties with language and cultural differences, and others who had no main role, such as having two or more roles with an equal score, were eliminated. The subjects started their first assignment by filling out the Self-Perception Inventory (SPI). Look at Appendix A for detailed information regarding the experiment's team composition and constraints.

4.1.3.1.2 Results and Conclusion

Task B is the only task that has significant differences in means, which suggests that there is a correlation between the grade of Task B and the total number of Completer-Finishers. Thus, the CF had a substantial effect on Task B. Also, only nine teams out of 18 got scores less than 40%. However, only one team out of six teams which had CF got a score less than 40%. In other words, around only 17% of the teams with Complete-Finishers got grades less than 40%, and around 67% of the teams without Completer-Finishers got grades less than 40%. It is obvious that Task B was harder than Task A since some students seemed to be confused about implementing the required algorithm. On the other hand, ten points were distributed among any team who completed subtasks of task B, in order to get partial credits, even if they did not implement it successfully. Therefore, the role of Completer-Finishers showed positive effects on the projects' progress since they were motivated by goals, worked hard even if other team members were not doing their jobs, and tried to work to their best abilities to finish their tasks on time and earn as many points as they could.

Prior studies (Stevens 1998) demonstrate that the leadership roles of both of the chairman and shaper have a significant effect on software development team (This will be discussed later in this chapter). In contrast, the leadership role is not so important for teams who have only three members, as proven in this study. The Completer-Finisher role is the only role that has a positive effect on the software projects. It was concluded that not all the roles of the Belbin model could be applied to software team configuration. In other words, some roles may have significant effect on the software development while others may not, and that also depends on the nature of the tasks.

4.1.3.2 A Study that Applies the Roles of Shaper, Plant, and Monitor-Evaluator on Software Teams (Stevens 1998)

4.1.3.2.1 Research Hypotheses

The main focus of this research is to explore the importance of a software team that consists of at least one of the Belbin roles. More specifically, this study examined only three roles from the Belbin model that are most relevant to software development teams: the Shaper, the Plant, and the Monitor-Evaluator. Three null and alternative hypotheses were identified as follows:

1. Shaper: Leadership

   Null Hypothesis: Teams consisting of only one leader could perform equally to teams without a leader or with more than one leader.

   Alternative Hypothesis: Teams consisting of only one leader do not perform equally to teams without leader or with more than one leader.
2. Plants: Innovation

   Null Hypothesis: Teams consisting of all Plant individuals could perform equally to teams without one or more Plants.

Alternative Hypothesis: Teams consisting of all Plant individuals could not perform equally to teams without one or more Plants.

3. Monitor-evaluator: Decision Making

   Null Hypothesis: Teams consisting of all Monitor-Evaluator individuals could perform equally to teams without Monitor-Evaluators.

   Alternative Hypothesis: Teams consisting of all Monitor-Evaluator individuals could not perform equally to teams without Monitor-Evaluators.

The dependent variable that has been investigated for the three experiments is team effectiveness and performance. Thus, a quantitative analysis was performed in order to evaluate the performance of the teams by measuring the required time to acceptably complete a programming task as well as the time that is needed to come up with a solution that produces correct output. The mean completion time of teams expected to succeed is compared with the mean completion time of teams expected to fail in order to discover if there is a significant difference between the means.

4.1.3.2.2 Participants, Experiments, and Quantitative Results

24 students were distributed into eight teams; each team consisting of three members. Moreover, all the teams were balanced by the degree of programming skills. The participants were supposed to work on a single computer and use whatever editors, compilers, and utilities they preferred, as well as email their output to the instructor once they were done with their solutions. After that, a completion time was recorded, only if the output is correct. Each team was located in a group in order to test the different hypotheses based on the successful conditions and the unsuccessful situations. The groups were compared to explore if one group had a better performance due to the mean time to completion.

The leadership experiment shows that teams with a leader perform better than those without a leader: teams without leaders tend to spend more time solving the problems.

As a result of analyzing the data of the leadership experiment using ANOVA, it is shown that there is a significant difference between the groups that have a single leader, and those without leaders or with multiple leaders. The p-value = 0.0068, the R-Squared = 0.7665, and the means are 101.81 and 76.63 for the two groups, with standard deviation of 57.97 and 54.54 respectively. As a result, the null hypothesis is rejected since the groups performed in different ways and the teams with a single leader had better completion time. On the other hand, teams with multiple leaders seemed to produce more conflict issues that reduced teamwork since they do not have an obvious leader.

In general, it was observed that some teams worked cooperatively together while others seemed to work independently. Furthermore, most leaders were listening to other team members to get good ideas and suggestions. Some leaders encouraged individuals to communicate effectively in order to complete their tasks, while others were unfocused and had difficulties in communicating and understanding the problem. Moreover, some leaders participated in the design and the coding.

It is concluded from the first experiment that the role of Chairman is different than the role of Shaper. The Chairman recognizes the strengths and weakness of individuals, and encourages all team members to produce to their best abilities using their experience and knowledge. On the other hand, the Shaper communicates regularly with all team members to get ideas and opinions in order to make solid decisions. Furthermore, Chairmen seemed to be introverted and thoughtful, while Shapers seemed to be extroverted and motivated.

The innovation experiment shows the value of innovative Plant individuals to a software team. This study consists of three groups: All Plants, Some Plants, and No Plants. Individuals were distributed to the different teams based on their Computer Science GPA, in order to get balanced teams. The teams are arranged according to the number of plants they involve. Some teams are arranged within the Some Plants group since these teams consist of one or two plant members.

As a result of this experiment, teams consisting of all Plant individuals had better performance than teams composed of some plants. Generally, the mean time to completion for the All Plants group is significantly different than the Some Plants group. However, there was not enough evidence to differentiate between No Plants group and the other two groups. The means are 100.75, 86.25, and 54.13 with standard deviations of 55.81, 56.77, and 26.43 respectively for the following groups: Some Plants, No Plants, and All Plants. Moreover, the groups are statistically different (p=0.0412), and (R-Squared = 0.5024) in which, the null hypothesis is rejected since the groups are not similar to each other.

It was observed that teams with all Plant members tend to think more in a similar way when do planning and solve problems. Those people are innovators: they always start brainstorming and come up with useful ideas and suggestions.

The decision-making experiment illustrates the importance of recruiting Monitor-Evaluator individuals to a software team, in order to recognize the capabilities of individuals, evaluate alternatives and methods, select appropriate approaches, and make critical decisions. This study consists of two groups: Decision, and No Decision. Members were distributed to the different teams based on their computer Science GPA in order to get balanced teams. The teams are arranged based on the real existing of a Monitor- Evaluator. In other words, a team should have at least one obvious Monitor-Evaluator to be considered within the Decision group. Otherwise, it would be considered within the No Decision group.

It is logical to think that every software team should consist of at least one Monitor-Evaluator to make critical decisions. However, the statistical analysis of this experiment shows that there is not enough evidence that teams with Monitor –Evaluators perform better than teams without the decision makers. The p-value = 0.2269 and therefore, the null hypothesis failed and was rejected. This means that it could be possible that the role of Monitor-Evaluator is not necessary for software teams. On the other hand, it was observed that the first two teams that finished solving their problems correctly were from the Decision group, and the last two teams completed their assignments were from the No Decision group. Thus, more studies should be conducted to study the effect of this role on software development teams by either increasing the sample size or stating different assumptions.

4.1.3.3 Other Studies that Apply Belbin Theory on Assembling Teams

In this subsection, we examine some studies that applied Belbin model on assembling teams for the ACM programming competition, and the object oriented software design class. We also examine qualitative analysis of Industrial teams that applied Belbin role theory.

4.1.3.3.1 ACM Programming Competition (Stevens 1998)

This observational study provides a subjective analysis of data gathered from the ACM programming competition in 1996 in order to examine the structure of software development teams based on the Belbin theory. For the programming competition, 93 teams were formed; every team consisted of at most three students. Team members were limited to different constraints such as: they could bring only books, manuals, and program listings but they could not bring machine-readable versions of software or data. Furthermore, each team could use a single computer, could only use Pascal, C, and C++ as programming languages, and could exchange information only with their team members. Team members were asked to complete eight programming assignments correctly and then submit it to an evaluator for acceptance or rejection. In general, teams were arranged according to the number of problems they solved and the time of completion. The total time is the sum of the time spent solving the problem, submitting the solution, and getting acceptance, plus 20 minutes' punishment for solutions that are rejected.

Only 33 teams out of 93 provided complete Belbin data that had been analyzed by comparing two groups of teams. The first group represents seven teams that completed five to eight problems and are ordered in the top eighteen. However, the second group represents nine teams that completed at least one problem correctly and are ordered at the bottom of all teams, more specifically, from 68th to 93rd. The main qualitative differences between the two groups are:

1. Each group has a different set of incomplete roles: the average number of empty roles in the first group is les than 2.0, and the average number of empty roles in the second group is almost 3.5.
2. Every team in the first group has a single leader except one. However, all the teams in the second group have either multiple leaders or no obvious leader.
3. Every team in the first group has at least one Company Worker. In contrast, only 44% of the teams in the second group include of Company Workers.
4. 71% of the teams in the first group involve a Resource Investigator, while 22% of the teams in the second group contain a RI.
5. 71% of the teams in the first group involve a Team Worker, while 33% of the teams in the second group contain a TW.

This study shows the importance of involving different Belbin roles in software development teams, especially the roles of Company Workers, Resource Investigators, and Team Workers, as well as having a single leader for each team. In other words, Company Workers have essential roles in translating the project plans and concepts into practical procedures and methods that would definitely improve the progress of the teams. Furthermore, Resource Investigators are important to any software team since they have the ability to explore new things and useful ideas. Team Workers are also beneficial to the entire team members since they negotiate within the team, encourage individuals to work as one group, and ensure that everything is under control. Additionally, as it was observed in the previous section, a team with a single leader demonstrates better performance than teams with multiple or no leaders.

4.1.3.3.2 Object Oriented Software Design Class (Stevens 1998)

This experiment applies Belbin role theory to the second project of an Object Oriented Software Design class at Virginia Tech. This research asks the students to voluntarily provide data by filling out a Keirsey Temperament Sort (will be discussed later in this Chapter) and a Belbin Self-Perception Inventory. Three groups of 26 teams were constructed: the first group consisted of teams that were assembled based on balanced

Belbin roles; the second group involved teams that were gathered randomly since they did not provide Belbin data; and the last group included teams that were created by students who worked previously together but without considering Belbin roles.

The ANOVA result demonstrates that teams from the first and the second group have significantly different means, with respect to their project grades. Moreover, the number of Completer-finishers in a team created an obvious distinction between the groups. Hypotheses are stated in order to compare the different groups and test if the groups have similar project grades. Thus, the first hypothesis is:

Null Hypothesis: Teams that have worked together demonstrate similar performance as teams that are structured based on Belbin roles and those which are formed randomly.

Alternative Hypothesis: Teams that have worked together do not demonstrate similar performance as teams that are structured based on Belbin roles and those which are formed randomly.

The means are 96.657 for the teams structured based on Belbin roles, 91.164 for the random teams, and 94.623 for the self-selected teams with standard deviations of 3.34, 5.31, and 5.64 respectively. Also, F-test is applied (p-value is equal to 0.1182) so that the three groups are not statistically different and the null hypothesis is failed to be rejected. Following that, another hypothesis is stated but this time concerning only the first two groups, since it is not fair to compare them with teams that are familiar to each other.

Null Hypothesis: Teams constructed by experimenter based on Belbin roles demonstrate similar performance as teams that are formed randomly.

Alternative Hypothesis: Teams constructed by experimenter based on Belbin roles do not have similar performance as teams that are formed randomly.

The means are also 96.657 for the teams structured based on Belbin roles, and 91.164 for the random teams, but p-value for the F-test is equal to 0.0264. Thus, the null hypothesis is rejected and there is significant evidence that teams constructed using Belbin roles perform better than those structured randomly.

Another assumption was investigated based on the presence of the several Belbin roles by examining the effect of Completer-Finishers.

Null hypothesis: Teams involving Completer-Finishers perform better than teams without Completer-Finishers.

Alternative hypothesis: Teams involving Completer-Finishers do not perform better than teams without Completer-Finishers.

Generally, three types of groups were found: no CF, one CF, and two CFs, but no teams with all Completer-Finishers. As a result of ANOVA analysis, the groups were statistically different by F-test (p-value = 0.0441). The means were 84.0, 92.941, and 93.333, respectively. Another test was also applied on this data (Duncan's New Multiple Range Test (Ott 1993)) which showed that the means of the Group with no Completer-finishers were significantly different than the other two groups. Thus, teams with completer-Finishers performed better than teams without Completer-Finishers.

On the other hand, a qualitative analysis of the data showed important results after ordering the project grades of the 26 teams from the highest to the lowest. The first ten teams from the top had a single strong leader, while six teams in the lower half had a single leader. However, only one team in the lower half had all strong leaders. The bottom teams involved six Monitor-Evaluators with a score of at least 14 while the top teams included only two Monitor-Evaluators. The top teams consisted of nine different roles with a score of at least 20, while the bottom teams included only four roles. Additionally, the top seven teams were compared to the bottom seven teams: six of the seven top teams had a strong leader with a score of at least 14 while only two teams in the bottom had a strong leader. Furthermore, only one of the top seven teams had a solid Monitor-Evaluator with a score of at least 14, while six of the seven bottom teams had Monitor-Evaluators. Therefore, the leadership role is very essential to software team success. However, Monitor-Evaluators may not be useful for software development teams, or only one or two of them might be desirable as implied by previous studies.

This study also provides some statistical data descriptions for the Belbin roles. The means are 6.93, 9.84, 9.28, 6.38, 8.51, 11.67, 8.51, and 7.85 respectively for the following roles: Chairman, Shaper, Plant, Resource Investigator, Monitor-Evaluator, Company Worker, Team Worker, and Completer-Finisher. If the roles were distributed equally, then the mean would be 8.75. It is obvious that software programmers do not prefer to be Chairmen, Resource Investigators, and Completer-Finishers. In contrast, they prefer becoming Shapers, Plants, and Company Workers.

4.1.3.3.3 Qualitative analysis of Industry Teams that Implemented Belbin Role Theory (Stevens 1998)

This study applied Belbin role theory to industrial software teams. The main concern of this research was to examine the six problem areas that Belbin (Belbin 1981, 1996) specifies:

1. Determining goals, objectives, and needs.
2. Innovation and discovering thoughts.
3. Decision-making and planning.
4. Meeting, making contacts, and exchanging information.
5. Translating plans into procedures and building organizations.
6. Paying attention to details.

The experimenter formed several teams; each of consists of three to nine members. The individuals filled out the Belbin Self-perception Inventory to provide data for analysis. The experimenter also conducted some meetings with the teams' leaders to get independent analysis, analyzed the self-perception data looking for the data trends, and made a comparison between the Belbin analysis and the interview information determining similarities and differences.

The first qualitative analysis was performed on a software development team working on an internal Hardware-Software Interface (HSI) program at a telecommunication company. The project team consisted of four members with a project manager that controlled the team, designed the hardware, and assisted individuals in completing their tasks. On the other hand, the first three members were experienced enough in software development while the fourth member was working as a part time consultant in order to provide domain knowledge for the rest of the team. The following analysis investigated the team without having Member 5 as a manager.

The team consisted of an extraordinary indication of plants who developed new methods, approaches, and ideas with particular attention to the most important aspects, and searched for possible breaks in approach to the problems with which the group was confronted. Members 1 and 3 seemed to have taken the lead in the absence of the manager, following the roles of Shaper and Chairman. Member 1 scored almost equally in all of the roles, which means that this individual had no clear single role. However, Member 3 had a stronger indication of being team worker as the primary role and performed the role of chairman and plant as the secondary role. Thus, neither of these two members took the leadership role. Member 4 shows a conflict because of which, it was impossible for this part-time individual to be strongly involved as a Company Worker.

As a result of the data analysis and interviews, the team should have produced innovative ideas and suggestions because of the large number of Plant members. Member 2 was a strong Completer-Finisher which indicates that the team included an individual who had the ability to get the work done on time. Member 5 was shown to be important as a single, strong project manager who could effectively lead the entire team, since no other team members demonstrated sufficient ability to focus on leading. Furthermore, member 5 could fill the role of company Worker when necessary.

The second qualitative analysis was performed on software the Regional Team working on to develop a document management application. The project team consisted of nine members: the first two members were software developers who shared the roles of project manager and technical leader. The two developers started producing a prototype, but after six months, they found that they needed more team members who had particular experiences in adding features and fixing bugs, in order to obtain the final release of the product and meet the deadlines.

Some teams involved a large number of Plant members who basically provided innovative ideas and suggestions and improved the team's success. Additionally, this team was full of Company Workers but satisfied the Belbin role complement, except for the Completer-Finisher role. This had the potential to cause some problems such difficulties meeting deadlines and less attention to detail. The team could have had multiple leaders since three members acted as Shapers, and only one acted as Chairman. However, the role of Monitor-evaluator seems to have disappeared in this team.

As a result of the data analysis and interviews, the team members focused on coming up with new and innovative ideas more than interacting within each other while solving particular problems. Member 7 was a strong Shaper and was considered the source of the problem since he/she was given freedom to design in his/her own way without following the team manager's direction. Some members worked hard filling the gaps, and solved conflicts as much as they could, such as Members 3, 5, 6, and 8. Moreover, the leaders improved the team performance by allowing individuals' brainstorming and innovative thinking. Some members were excluded from acting as team leaders for many reasons: they were new to the teams, were not obvious leaders, and were not native English speakers. A Resource Investigator is selected according to the level of interaction with people outside of the team: the more interaction an individual had, the better Resource Investigator he/she was. The two leaders of this team acted perfectly since they came up with effective decisions. However, one of the leaders was also a Monitor-Evaluator which was apparently an important role for this experiment, which was the opposite of the conclusion from the previous case studies. The Company Workers were useful in turning plans into reality. However, there were some important aspects that were missing, and other problems that occurred, because of the absence of the Completer-Finisher's role. In general, although there were some significant problems such as lack of details, team members expressed satisfaction with their work and their product.

The third qualitative analysis was performed on a team consisting of individuals from the Design Studio seminar class. The students were supposed to use knowledge from their education in software systems engineering, software design, and project management in a real world project as part of their software engineering master's degree. During one academic year, the students developed a Windows-based project management tool that integrated Albrecht's Function Point Analysis with Barry Boehm's COCOMO model. The team successfully produced 8,000 lines of C++ code, prepared design specification documents, evaluated plans, documents guides and programs, and provided help features.

This team demonstrated tremendous and well-balanced Belbin roles among the team members. However, there was only one obvious strong Plant member and another individual whose third role was considered as a weak Plant. There are no Monitor-Evaluators except one individual who was already acting as a project leader. Additionally, the role of Completer-Finishers was also missing in this team since only one member had it as a third role. It is also shown that there was problem with the leadership role since three members indicate the Shaper role.

As a result of the data analysis and interviews, the lack of Plant members may have caused a serious barrier to the success of the team, since the innovation and discovery of new technologies might have been missing. Moreover, Plants have the ability to find solutions to problems, and reengineer existing products to new applications. Thus, the low amount of Plant members could have negatively affected the performance of the team. It was observed that Member 5 was the technical leader, who was responsible for presenting designs and several ideas that could be integrated into the final product. Members 3, 4, and 5 had the ability to do testing and make critical design decisions, but Member 2 was useful in documenting and helping other team members resolve conflicts. Member 3 was known as the COCOMO expert, while Member 4 was known as the Function Point specialist. The design process was documented successfully since this team consisted of good designers, especially Members 3, 4, and 5. In general, the teams performed very well in terms of meeting the short-term goals. However, the team suffered from having individuals who had a lack of practical experience and knowledge; they did not realize how time, effort, and cost could affect the long term objectives. Furthermore, the chairman role was almost missing in this team; this could have affected the entire team by not providing a general view of the project. The team also suffered from not being able to make contacts since the role of resource Investigator was not defined well.

The fourth qualitative analysis was performed on a software development team from a large manufacturing company which delivers a set of products with standard specifications by adding more features or modifying exiting elements, based on the customers' needs. Most of the company's projects are written in Pascal and C languages. One of the company procedures is that the team leader is the one who acts as the technical leader and project manager and forms the whole team based on their skills and capabilities.

It is also shown that there was only one obvious leader on the team, which counts as a positive feature. The team consisted of two strong Company Workers because they believe that Company Workers can prevent strife between individuals, remind others not to waste time and effort on trivial problems, and play the role of Plant members exploring innovative ideas. All Belbin roles are distributed among team members except the roles of Completer-Finisher and Plant. Thus, this could have affected the factor of innovation of the entire team and raised the problem of less attention to detail.

As a result of the data analysis and interviews, Member 1 was the technical leader and project manager of the team. In addition, Member 1 worked collaboratively with Member 2 on the process of building the software. Member 4 was responsible for verifying the processes and the simulator in order to satisfy the customer's requirements, while Member 4 was an integrator who assembled the small pieces of code and tested them to ensure their quality and functionality. This team was controlled effectively by the team leaders and was successful in meeting the short-term goals. However, focusing on the long-term objectives was missing among the manufacturing team members. This experiment also shows that the existence of a single leader is vital since he/she provides more details, recognizes individuals' abilities, and knows how to organize tasks based on their priorities.

The fifth qualitative analysis is performed on a medium software development team from an industrial company that produces large software systems. This project produced "a large Windows 3.1 application that provides an interface to control industrial machinery". The team consists of three members who were working together for the first time but each one had enough experience working previously on such kind of development projects.

The team had neither Plant members nor Monitor-Evaluators. This could have affected both the evaluation and innovation factors and led the team not to explore new technologies, find creative solutions, or evaluate approaches and alternatives to make balanced decisions. However, all the team members were Company Workers, which means that they were willing to work hard and achieve something. Thus, the role of Plant may not have been that essential since all individuals were Company Workers. The team also involved two leaders who cooperatively complemented each other: one played the role of Chairman and the other acted as a Shaper.

As a result of the data analysis and interviews, team members stated that they did not have an obvious leader. All team members understood the short-term goals very well but had difficulty recognizing the long-term objectives. The team rated very high in meeting each other, making contacts, exchanging information, and paying attention to detail. On the other hand, the team rated very low in innovation and discovery, as well as translating plans into procedures and building organizations. The team rated fairly in determining goals, objectives, and needs as well as making decisions and developing plans.

4.2 Myers-Briggs Model and Keirsey Theories

This model focuses on measuring the personality characteristics of individuals. Myers-Briggs Type Indicator (MBTI) is a personality test that determines the behaviors and activities of people by measuring four different aspects of personality that are totally independent of each other. In contrast, the Keirsey Temperament Sorter is a more recent tool that provides useful information related to people's personalities based on the same concept of MBTI. Generally, both of the instruments specify personal preferences, produce results in an identical format, and help researchers collect data in an effective way. Additionally, Stevens illustrated significant statistical correlations between the MBTI and the Keirsey Temperament Sorter when applied to software development teams (Myers 1985).

The four personality aspects of the Myers-Briggs Type Indicator and Keirsey Temperament Sorter are Introversion-Extroversion (I-E), Intuition-Sensing (N-S), Thinking-Feeling (T-F), and Perceiving-Judging (P-J). The following section discusses the four scales of MBTI in more details.

4.2.1 MBTI Four Aspects (Gifford 2003)

1. Energizing: Extraversion-Introversion (E-I)

The first aspect deals with how an individual gets his/her energy to do things and get things done. In other words, it focuses on how a person pays attention to the outer world, extracting things and information from people, versus a person who just concentrates on suggestions and impressions that exist in the inner world.

Extroversion (E): is the behavior that describes a person who is more interested in what goes on around than in personal thoughts and feelings. Thus, extroverts are sociable and enjoy obtaining energy from the outside world of people and activities.

Introversion (I): is the behavior that describes a person who is more interested in his/her own thoughts and feelings than in things outside. Thus, introverts prefer working alone, getting energy from their internal world of suggestions and impressions.

2. Attending: Sensing-Intuition (S-N)

The second aspect deals with an individual style of collecting information: some people gather data based on their senses as facts, details, and precedents while others gather information indirectly as relationships, patterns, probabilities, and possibilities.

Sensing (S): is the behavior that describes a person who prefers using the senses to discover what is real.

Intuition (N): is the behavior that describes a person who prefers using the imagination to visualize what could be possible and “looks beyond the five senses”.

3. Deciding: Thinking-Feeling (T-F)

The third aspect deals with an individual style of making decisions: some people prefer making their decisions based on objectives and impersonal logic while others prefer making their decisions based on “person-centered values”.

Thinking (T): is the behavior that describes a person who prefers to arrange and categorize information in a logical and objective manner in order to make decisions.

Feeling (F): is the behavior that describes a person who prefers to arrange and categorize information in a personal manner in order to make decisions.

4. Living: Judging-Perceiving (J-P):

The fourth aspect deals with an individual lifestyle: some people prefer having everything planned and structured, while others are flexible and prefer being open to alternatives and choices rather than deciding.

Judgment (J): is the behavior that describes a person who prefers living a planned and structured life.

Perceiving (P): is the behavior that describes a person who prefers living in unplanned, unstructured, and flexible life.

4.2.2 The sixteen Types of Personality (Gifford 2003)

Sixteen types of personalities result from having different combinations of the four main aspects of Myers-Briggs personality types. Each personality type consists of four letters corresponding to the first letter of each of the main four personality aspects. For example, ISTJ stands for an Introverted Sensing Thinking Judging person.

1. ESTJ

The ESTJ person obtains energy from what goes on in the outside world, extracting information from people and getting important things via social activities. This person also prefers dealing with facts, details and procedures and makes decisions based on objective and impersonal logic. Additionally, this person is structured and organized in his/her life. Thus, this person is more practical, focuses on details rather than basic ideas and methods, and prefers finding particular solutions that have been evaluated successfully or used effectively in prior projects to ensure high quality results.

2. INFP

The INFP person obtains energy from concentrating on thoughts, suggestions, and impressions that exist in their inner world. This kind of people prefers using his or her imagination to understand something without conscious reasoning or study, and makes decisions based on his/her personal values. This person is flexible enough in his/her life to try new approaches and is open to accepting different possibilities as they appear. Therefore, this person is quite adaptable, seems to be interested in new ideas and thought, makes innovative contributions, works on serious tasks that have clear objectives and scopes, and prefers to excel and improve by him/her self without getting help from others.

3. ESFP

The ESFP person obtains energy from what goes around in the outside world, extracting information from people and getting important things via social activities. This person also prefers dealing with facts, details and procedures, and makes decisions based on his/her personal values. Furthermore, this person makes friends easily and enjoys getting together with other people to discuss what is happening in current situations and to propose ideas. This person is friendly, impulsive, and flexible enough to get pleasure from living his/her daily life, and has the ability to respond to issues as they arise. Additionally, this person participates effectively in solving critical problems, such as troubleshooting, and cooperates successfully with others in practical circumstances.

4. INTJ

The INTJ person obtains energy from concentrating on thoughts, suggestions, and emotions that exist in their inner world. This kind of person prefers using imagination to understand things in the future without conscious reasoning or study, and makes decisions based on his/her impersonal and logical analysis. Additionally, this person is structured and organized in his/her life. Thus, this person is steady, has a strong intellect, and has the ability to determine long-term objectives and goals. Moreover, this person seems to be skeptical and critical, not believing claims and statements that have not been logically approved.

5. ESFJ

The ESFJ person obtains energy from what goes around in the outside world, extracting information from people and getting important things via social activities. This person also prefers dealing with facts, details and procedures, and makes decisions based on personal values. Additionally, this person is structured and organized in his/her life. Thus, this person is sympathetic and warm-hearted, takes responsibilities, shows loyalty, enjoys serving people, looks to improve relationships with colleagues, and makes useful contacts with new friends. However, this person has difficulty handling conflicts and criticism.

6. INTP

The INTP person obtains energy from concentrating on thoughts, suggestions, and emotions that exist in the inner world. This kind of person prefers using imagination to understand future events without conscious reasoning or study, and makes decisions based on objective and impersonal logic. Additionally, this person is flexible enough in his/her life, going with new approaches and open to accepting different possibilities as they appear. Thus, this person is quiet, adaptable, and detached, not easily influenced by personal opinion or by others. This person looks to improve procedures and makes effective changes and enhancements to the daily routine and uses his/her intellect ability to solve complex problems.

7. ENFP

The ENFP person obtains energy from what goes on in the outside world, extracting information from people and getting important things via social activities. This kind of person prefers dealing with patterns and possibilities and using imagination to understand future events without conscious reasoning or study. This person also makes decisions based on his/her personal values. Additionally, this person is flexible enough in his/her life, accepting new approaches and being open to accepting different possibilities as they appear. Thus, this person is innovative and insightful, seeking out new solutions and useful ideas. Furthermore, this person enjoys working collaboratively with colleagues towards achieving the organization’s goal. However, this person has a tendency to ignore some details and planning.

8. ISTJ

The ISTJ person obtains energy from concentrating on thoughts, suggestions, and emotions that exist in the inner world. This person prefers dealing with facts, details and procedures and makes decisions based on a logical basis. Additionally, this person is structured and organized in his/her life. Thus, this person is quiet, serious, practical, and has a clear vision that helps him/her understand the various situations and visualize the future's circumstances.

9. ESTP

The ESTP person obtains energy from what goes around in the outside world, extracting information from people and getting important things via social activities. This person prefers dealing with facts, details and procedures and makes decisions based on objective and impersonal logic. Additionally, this person is flexible enough in his/her life, open to new approaches and willing to accept different possibilities as they appear. Therefore, this person is practical, impulsive, excels in troubleshooting and tries to use his/her best ability to solve complex problems. However, this person sometimes does not finish particular tasks by him/her self since he/she works best with others.

10. INFJ

The INFJ person obtains energy from concentrating on thoughts, suggestions, and emotions that exist in the inner world. This kind of person prefers using imagination to understand future events and possibilities without conscious reasoning or study and makes decisions based on his/her personal values. Additionally, this person is structured and organized in his/her life. Thus, this person understands the purpose of life and works in a regular manner to accomplish his/her goals. This person likes to help people and convey his/her knowledge and skills to others but without showing that directly.

11. ENFJ

The ENFJ person obtains energy from what goes on in the outside world, extracting information from people and getting important things via social activities. This kind of person prefers using imagination to understand future events and possibilities without conscious reasoning or study, and makes decisions based on his/her personal values. Additionally, this person is structured and organized in his/her life. Thus, this person is highly sociable, working best when with other people, looking to develop stable relationships with others, and helping them improve their personal skills and capabilities. However, this person may have difficulty criticizing or expressing his/her feelings to friends in order to avoid damaging long-term relationships.

12. ISTP

The ISTP person obtains energy from concentrating on thoughts, suggestions, and emotions that exist in the inner world. This person prefers dealing with facts, details and procedures, and makes decisions based on logical basis. Additionally, this person is flexible enough in his/her life to try new approaches and is open to accepting different possibilities as they appear. Thus, this person is impulsive, quiet, adaptable, and detached; he/she is not easily influenced by personal opinion or by others. Furthermore, this person is good at discovering unexpected new ideas and solutions.

13. ENTJ

The ENTJ person obtains energy from what goes around in the outside world, extracting information from people and getting important things via social activities. This kind of person prefers using imagination to understand future events and possibilities without conscious reasoning or study, and makes decisions based on an objective and logical basis. Additionally, this person is structured and organized in his/her life. Thus, this person seems to have good experience in dealing with life, managing people to fulfill goals. This person also prefers working with tolerant people who can take responsibility and perform their work successfully.

14. ISFP

The ISFP person obtains energy from concentrating on thoughts, suggestions, and emotions that exist in the inner world. This person prefers dealing with facts, details and procedures and makes decisions based on personal values. Additionally, this person is flexible enough in his/her life to try new approaches and is open to accepting different possibilities as they appear. Thus, this person is quiet, adaptable, supportive, and friendly, cooperating effectively with people and constantly helping them.

15. ENTP

The ENTP person obtains energy from what goes on in the outside world, extracting information from people and getting important things via social activities. This kind of person prefers using imagination to understand future events and possibilities without conscious reasoning or study, and makes decisions based on an objective and logical basis. Additionally, this person is flexible enough in his/her life to try new approaches and be open to accepting different possibilities as they appear. Thus, this person is creative, adaptable, focuses on new thoughts, makes valuable changes, and tries to find most recent applicable solutions to different problems.

16. ISFJ

The ISFJ person obtains energy from concentrating on thoughts, suggestions, and emotions that exist in the inner world. This person prefers dealing with facts, details and procedures and makes decisions based on personal values. Additionally, this person is structured and organized in his/her life. Thus, this person is quiet, serious, and seeks to make useful relationships with colleagues. Moreover, this person enjoys serving people, shows respect, and knows how to criticize individuals in an appropriate way that could not cause harm to their feelings.

4.2.3 Previous Studies

This subsection outlines a number of previous studies related to Myers Briggs model. It also examines the effect of building software teams using the MBTI, the personality impact on IT team projects, and the effect of personality type on team performance.

4.2.3.1 Building Effective Software Teams Using MBTI (Gorla 2004)

Poor software project team structure is one of the major factors that could affect the entire team's performance, cost, and schedule. There are other important aspects that could lead to the same poor results if completely ignored while composing software teams: the personality of individuals, team leadership, team communication and coordination. Not many experiments have investigated the combination of individual personalities within a software project framework. However, White (White 1984) examined two teams in a real world field, studying only two personality dimensions, and discovered that winning teams have various kinds of personalities. Furthermore, Yellen (Yellen 1995) found that "extroverts are relatively more effective than introverts in group decision making."

This study used the MBTI to evaluate every team member's personality style in 20 small software development teams in Hong Kong and explored the relationship between the personality composition of teams and the entire teams' performances. More specifically, this research identifies the influence of a project leader's personality on team performance, the effect of team members' personalities on team productivity, and the influence of the heterogeneity of personalities on team performance. This research also focuses on four scales: social interaction, information collecting, decision making, and external world communication. Each scale includes two feasible personality types. The social interaction scale organizes an individual as either an extrovert (E) or an introvert (I). The information collecting scale arranges an individual as either sensing (S) or intuitive (N). The decision making scale classifies an individual as either thinking (T) or feeling (F). The last scale, which measures the person's ability to communicate with the external world, is categorized as either judging (J) or perceiving (P).

4.2.3.1.1 Participants and Experiment Procedures

92 IS professionals from 20 software teams in Hong Kong were asked to participate in this study, completing the MBTI questionnaires and providing their opinions about team productivity. Only 97% of the IS professionals responded where every team consists of three to seven members, with an average of four people. In general, 57% of the team leaders between the ages of 31 and 35 had computing experiences working in the field for more than 10 years; 43% have advanced degrees. However, 60% of the team leaders between the ages of 26 and 30 have worked for at most 5 years; only 17% of them have earned academic degrees.

The different personality types were measured using the Keirsey Temperament Sorter, asking the participants to answer 70 questions related to their response to the various circumstances. Additionally, a comprehensive performance tool (Jiang 1997) was used in this study in order to measure six performance criteria. For example, the quality of work was measured by asking the participants to rank what they feel towards the work quality from 1, which represents low quality, to 5, which represents high quality.

4.2.3.1.2 Results and Conclusion

The most effective personality attributes for each software development team role, found as the result of this research, software teams with intuitive (N) leaders perform better than those with sensing (S) leaders in information gathering. Intuitive leaders look at the whole picture of both the systems and subsystems, have the ability to come up with innovative and creative ideas, and can easily find alternative solutions for complex problems. This characteristic is essential for a software team leader, especially if he/she also gets involved in system analysis. The results also show that team leaders with a feeling (F) type perform better than those with the thinking (T) type in making decisions. Feeling leaders are open-minded, and arrange and categorize information in a personal manner in order to make decisions. It is observed that feeling leaders collaborate effectively with individuals and encourage them to express their ideas, since this kind of person is willing to consider constructive suggestions and recommendations from all team members.

It is shown that the decision making factor (T/F) of the system analyst's personality has a significant effect on team productivity. In other words, teams consisting of thinking (T) type system analysts perform better than those with the feeling (F) type. Thus, thinking system analysts arrange and categorize information in a logical and objective manner in order to make decisions. It is also observed that these kinds of people have the ability to work on multiple analytical tasks; this allows them to participate successfully in the system design and programming phases. On the other hand, the social interaction factor (I/E) of the programmer's personality has a strong influence on team performance. Teams with extroverted (E) programmers perform better than those with the introverted (I) personality type. This means that programmers need to be friendly and sociable in order to interact effectively with other software engineers, such as managers, leaders, systems designers, analysts, developers, operators, and other programmers. This interaction helps programmers understand the problems very well without confusion in order to develop appropriate solutions.

The heterogeneity of the team leader and team members is computed as the absolute difference between the score of the team leader and the average score of the rest of the team members. The factors that are related strongly to team performance are: social interaction (I/E) and information collecting (N/S): the higher the heterogeneity level between the team leader and team members, the higher is team performance and productivity. However, the level of heterogeneity of personalities among the team members has no significant impact on team performance, since the team sizes are small and most of the individuals are participating in multiple phases of the entire system development's life cycle.

It is concluded that software managers should assemble their teams based on the team members' personalities, since personality type analysis could help managers put together high-performance software project teams. Furthermore, it is observed that having diversity among team members is unnecessary because individuals may need to perform more than one task of the system development phases and such heterogeneity is not recommended for all life cycle phases. However, the previous condition is not applied to team leaders. Project managers should recruit intuitive team leaders for gathering information, since those leaders are forward-thinking in exploring future requirements. Additionally, project managers should select leaders with a feeling type to make critical decisions. Managers should recruit both leaders with the judging personality type in order to effectively deal with the external world, create milestones, and monitor the progress of all team members to achieve project goals.

Effective software teams should include individuals with the judging personality type in order to submit the final product on time without having any delay. It should also consist of system analysts with the thinking personality type in order to make decisions based on logical reasoning, methods, and approaches. However, system analysts with the sensing personality type are also needed in order to gather information, provide detailed design and programming specifications, and perform some programming tasks. Software teams also should involve extroverted people to encourage social interaction among the whole team. Teams should recruit programmers with the sensing personality type for gathering information and others with the judging personality type for dealing with the external world.

4.2.3.2 The Personality Impact on IT Team Projects (Peslak 2006)

“Information technology projects are almost always team efforts.” Personality is one of the most important factors that influence the way that teams achieve their objectives. Therefore, this research investigates the relationships between personality and IT team success. Individuals’ personality types are classified according to MBTI, while the success of teams is evaluated by measuring the teams’ outcomes. Furthermore, team processes are assessed using a comprehensive concept; the diversity of personality within a team is also examined. Additionally, this research was conducted since limited work has addressed the effect of personality on information technology projects.

4.2.3.2.1 Hypotheses, Participants, and Methodology

This research states three hypotheses that investigate the interrelationships between personality characteristics, team processes, personality diversity, and project success. The three hypotheses are listed below:

1. MBTI personality characteristics have a significant effect on the overall IT team processes.
2. MBTI personality characteristics have a significant effect on the overall IT project success.
3. Diversity in team personality composition has a significant effect on the overall IT project success.

55 Undergraduate students doing a semester-long term project formed 18 teams (each consists of 2-5 members) that participated in this research. They were asked to fill out a paper-based survey and classify themselves into each of the four personality categories (See Appendix B for more details regarding the personality descriptions). Team processes were evaluated via a number of questions relevant to team effort and performance, team roles, team building, leadership, communication, and conflict resolution. Gender and age determined the demographics studied in this research. Two-

thirds of the students were between the ages of 18 and 24 while others were between 25 and 30. Most of the students were male. The distributions of personality types are somewhat close to the means 1.5, which means that there is a relatively small indication that participants are introverted, intuitive, and thinking. However, judging and perceiving personality types are approximately located at the mean.

4.2.3.2.2 Results and Conclusion

After analysis of the eleven questions of team process through scale reliability and confirmatory factor analysis for every single team process dynamic, only one factor appears to have a value of 6.065 and all components have values more than 0.5; the reliability scale led to a very high alpha value of 0.9167. Multiple linear regressions are used to evaluate the effect of each of the four averaged MBTI team values on the team process dynamic. Also, all factors are not significantly correlated with the team process variable at p-value less than 0.1. Thus, the first hypothesis is rejected.

The results also show that personality characteristics play a vital role in improving team performance: three out of four MBTI classifications are shown to have a significant correlation with the dependent performance variable and project grade at p-value less than 0.1. In addition, the R-squared is 0.509, which is considered to be very high. However, both the gender and age seemed to be insignificant variables. It is concluded that extroverted, judging, and thinking people all together improve the success of the project, since the results are significant at p-value less than 0.10. Therefore, the second hypothesis is failed and rejected; the MBTI personality characteristics have significant influence on overall project success.

The third hypothesis was examined by calculating the coefficient of variation, which equals to the standard deviation divided by the mean value of a set of MBTI scales. Also, a regression analysis was done between the coefficient of variation independent variable and the project grade dependent variable. However, the correlation is not significant at p-value less than 0.10 in which the third hypothesis is rejected. Thus, diversity in team personality composition does not have a significant influence on the overall project success. Overall, this study does not show enough evidence that supports all the hypotheses, since the sample size is small and the experiments were not conducted in a real world industry. However, this study illustrates that MBTI personality characteristics have a significant effect on the overall IT project success.

4.2.3.3 A Study that Explores the Effect of Personality Type on Team Performance (Bradley 1997)

An effective team is a product of an appropriate team structure which involves increased motivation, higher task commitment, advanced levels of communication and collaboration, greater levels of performance, and the ability to resist stress, produce creative solutions, and decrease the development time. Additionally, productive teams benefit from effective leadership, intra-team communication, and group cohesion. Therefore, the study of personality is increasing in importance, since the characteristics of building productive teams depend at least to some degree on the personality types of team members. This study presents a model that highlights the influence of the personality type composition of a team on the entire team's performance and productivity. Furthermore, this research implements the MBTI theory in the team-constructing process and shows how it is important to consider the personality aspect when forming software teams by evaluating a case study of two software development teams.

The research starts with combining four major individual variation characteristics of building effective teams: leadership, communication, cohesion, and heterogeneity. This combination is considered as the base of forming an evaluative model of the effect of personality aspect on team productivity.

The leadership factor has a substantial impact on the team's overall success. Therefore, leaders should be not only knowledgeable and expert in group dynamics, but also be able to lead individuals who have different capabilities and personality types. Furthermore, team leaders should have the technical skills and appropriate personality types to enable them to conduct meetings, control and track members, mitigate risks, and resolve conflicts. Intra-team communication is the second factor that has a great effect on the overall team performance, since this aspect establishes the communication channels that enable team members to effectively interact, share knowledge, exchange information, and meet the project goals and objectives. Cohesion is another important issue that should be taken into consideration when assembling teams. It is defined as "a crucial ingredient in team effectiveness." Thus, a cohesive software team encourages individuals to work together as one team, help each other, and resolve conflicts. The heterogeneity of the personality types of software individuals is an essential factor that has an extensive effect on the team success, because having a balance of personality types of people with diversity of skills, experience, and knowledge would definitely improve the team's performance.

4.2.3.3.1 Research Model

After implementing the MBTI theory on the four main aspects of the team performance, the best personality for a team leader, depending on the nature of the task, is an ESTJ or an ENTJ. For example, intuitive leaders are essential for situations that require searching for new approaches and using the most recent technologies to solve complex problems. However, sensing leaders are required for enhancing systems that are already exist and that need new technologies added to them. Extroverted, sensing, and thinking people would satisfy the need for intra-team communication among all team members more than the introverted, intuitive, and feeling individuals. For example, extroverts are important for projects that require a high level of communication, but are not recommended for some critical situations that do not require that high level of interaction, such as interrupting others to express different ideas that cause team members to become confused. In addition, sensing people are able to structure their ideas and suggestions in a way that everyone can easily understand, while intuitive people seem to think in complex ways that confuse others. Furthermore, thinking people can make decisions and judgments, and express their feelings in a direct manner, while feeling people always try to avoid hurting other individuals' personal feelings.

The cohesion factor is typically influenced by thinking versus feeling. Thinking team members express their opinions of others' ideas in a way that could hurt others' feelings. However, feeling people try to do their best to keep a balanced harmony of feelings, interest, and opinions among all team members. The heterogeneity of a team represents the number of each personality type that is available on a particular team. Generally, each personality type has some benefits to provide to the entire team. Furthermore, harmonized teams can reach consensus earlier and get more innovative results. On the other hand, a team with a large extent of psychological homogeneity could produce a lot of problems.

4.2.3.3.2 Case Study

This section discusses a case study of two information systems teams that illustrated an observable difference in both performance and productivity. The two IS teams belong to a software development company in the Southeastern USA. This research began when the company management assigned a software project assignment to the two teams asking them to develop information systems of comparable complexity. The company management observed that the two software teams had different levels of performance and productivity. The first team spent around two years working on that particular project and delivered a moderate quality software product; afterwards, the user department asked the developers to perform major modifications and revisions, since they were not completely satisfied with the final product. However, the second team spent only one year in the development process, finishing their assignments effectively on time and within the schedule, and produced a high quality software system.

The company management monitored the two teams closely, observing the members' attitudes and behaviors, and evaluating their performance during the entire development life cycle. For example, the management noticed that the individuals in the first team were not organized and had difficulties understanding each other because of poor communication. Furthermore, the management gathered a set of descriptive data and did some statistical analyses in order to determine similarities and differences between the two teams. According to the psychometric evaluations of all team members, the average team compositions of the two teams were slightly different. However, there were big differences in the personality type of all team members. All participants were college graduates and the average age of both teams was roughly the same. The problem-solving skills for the first team and the second team were equal to 66.3 and 67.5 respectively.

Also, the p-value is greater than 0.65 (all values measured by the Watson-Glaser Critical Thinking Appraisal), which means that there is no significant difference between both teams in problem-solving skills. Additionally, the IQs for the first team and the second team were equal to 26.4 and 29.0 respectively, and the p-value was greater than 0.59. It is obvious that the difference in IQ was not statistically significant, since the second team illustrated a very slightly higher IQ on the Wonderlic Personal Test. Therefore, the personality types of all team members were analyzed in order to discover any potential difference between the two teams, since there were no variations between both of the teams in terms of demographics, difficulty of tasks, and fundamental ability levels.

An analysis of the Myers-Briggs Type Indicator composition of the two teams shows that there are some observable differences between the members of the two teams. Kroeger and Thuesen (Kroeger 1992) state that it is essential to have diversity and balance in the personality types of different group individuals. The second team has more balanced member personality types than the first team, which could explain why the second team was more successful and effective. For example, the first team involves 20% extroverts and 80% introverts, while the second team includes an equal percentage of 50% of both types. Kroeger and Thuesen also conclude that introverts are less communicative and always seem to keep information only to themselves without sharing it. It is quite clear that the reason the first team did not perform very well was that around 80% of its members were introverts.

Kroeger and Thuesen observe that sensing people prefer focusing on the details of their individual tasks more than looking at the whole picture or considering the major task of the entire teamwork. However, intuitive people like the teamwork concept, but have some difficulties translating that concept into practical actions and activities or integrating small pieces of subsystems into a larger system. Thus, a balance of sensing and intuitive members is needed in software development projects. The second team had an approximate balance in the type mixture of information gathering (S/N): it had around 57% intuitive and 42% sensing members. In contrast, the first team did not have a well-balanced combination of (S/N): it included 60% intuitive and 40% sensing individuals. The second team has also an approximate balance in the type mixture in the area of decision-making (T/F). The second team includes 42% feeling members while the first team consists of only 20% feeling individuals. This also explains the success of the second team, since feeling people care about ensuring that all team members are working properly together. Simultaneously, other thinking members focused on getting the job done and accomplishing the required tasks successfully. An obvious reason the first team was ineffective was that it included more thinking people than feeling, which resulted in the entire team focusing on just finishing their tasks, but without paying attention to users' needs and without caring much about the thoughts and suggestions of other feeling individuals.

Conversely, the first team had a better balance of judgers and perceivers: 70% of the team was judgers and 30% were perceivers. In contrast, all the members of the second team were judgers with an exact percentage of 100%. For this situation, Kroeger and Thuesen state that the variation in J/P is "the key to team success or failure." Judgers prefer to move from one step to another in order to stay on schedule and meet project objectives and deadlines, while perceivers consume more time considering other alternatives and issues. This could most likely explain why the second team delivered their product on time, while the first team had delays.

The leader of the second team was an ESFJ, while the leader of the first team was an INFP. It is obvious that three preferences distinguish the characteristics of the leaders of the two teams. The leader of the second team was more effective in leading the team to succeed since he/she was an extrovert, which may have made him/her more successful in encouraging individuals to communicate, share information, and get involved in every process. Furthermore, he/she was a sensing leader, which may have been more effective in keeping team members regularly on task. He/she was also a feeling leader, which may have been more effective in interacting usefully with a large percentage of feeling team members in order to support group harmony. In addition, the second team leader was a judger, which was in agreement with the other individuals' behaviors in pushing everybody to submit the final product on time. However, the leader of the first team was an introvert, which may have made him/her ineffective by refusing to give information and permission and trying to conduct very short meetings. The second team leader was intuitive, which may have caused him/her to be unsuccessful in putting concepts and theories into practical actions. The leader was considered as a feeling type who may have been ineffective since most of the members are thinkers. In other words, the leader may have focused on supporting group harmony rather than getting work done which could have frustrated the other individuals. Additionally, the leader was a perceiver, which may have made him/her more ineffective by wasting time considering other alternatives without focusing on the major tasks.

The second team facilitated better group communication since it involved a large percentage of extroverts, feelers, and judgers, while the first team had less successful group interaction and communication since it included a large percentage of introverts, thinkers, and perceivers. According to the data analysis, team composition of personality types appears to be an essential descriptive variable for variations in team performance. Generally, diversity as well as balance in individuals' personality types is required to assemble successful and productive software teams. Thus, the second team is more successful and effective since it involved a balance of extroverts and introverts, sensing and intuitive types, and thinking and feeling types, not to mention 100% judgers who ensured the completion of the project on time.

4.3 Other Models and Theories

Several team configuration models have been proposed by psychologists claiming that their models have specified effective methodologies and approaches for classifying members into teams based on their characteristics and behaviors, but only for general populations and purposes. No model or theory has been built specifically for software engineering projects, exploring their characteristics and traits in detail. Thus, previous models could partially benefit the process of assembling software teams, but not completely. In other words, no model or theory claims to be comprehensive, but at least some approaches as well as a subset of characteristics and behaviors related to software development could be found and applied successfully to software teams (Gifford 2003).

Belbin (Belbin 1981, 1996) states that the main purpose of any research related to team structuring with respect to psychology should be focused on exploring the sources of the variations between members on the team, and then studying the impact of those differences on team performance and productivity in order to produce an effective model.

Some models recommend building teams based on functional roles; other theories suggest configuring teams based on measuring the personality characteristics and traits. Therefore, combining both the functional roles and personality types in a team could produce a well-balanced team. “The success of the team depends on the interlocking pattern of the roles or types (Gifford 2003).”

Most personality theories are quite similar to each other in terms of evaluating identical characteristics and behaviors. Thus, those models “fit under the umbrella of the Five Factor Model” which theoretically specifies personality types according to five various perspectives (Costa 1992). MBTI (Myers 1998) and Keirsey theories are strongly related to the Five Factor Model since it involves the main four dimensions of personality types: Extraversion vs. Introversion, Sensing vs. Intuition, Thinking vs. Feeling, and Judging vs. Perceiving (Keirsey 1984). Additionally, 15FQ (Budd 1992) is another model that evaluates fifteen various personality scales, but it could also be summarized into five major personality aspects (Costa 1992).

On the other hand, other models and theories focus on roles such as Belbin model, the Team Management Index (TMI) (Margerison 1990), and the Management Team Roles Indicator (MTR-I) (Linda 2002). Generally, both the TMI and MTR-I depend strongly on the theory behind the MBTI, but simultaneously determine roles based on the Belbin model (Gifford 2003). In other words, the TMI and MTR-I models result from mixing both the MBTI theory and the Belbin model. However, every single model combines them in a different manner.

4.3.1 Management Team Roles Indicator (MTR-I)

The main idea of the MTR-I model is to build successful teams based on a particular set of roles, similar to the Belbin model. The success of a team results from the effective interaction between individuals and the various team roles. The major advantage of this model is to allow team members to evaluate the current situation of the team, instead of focusing on characteristics and behaviors. In other words, this model provides individuals the chance to identify what is currently happening to the team in order to reemphasize certain team roles. In general, no concrete work has been published implementing the

MTR-I in software team development (Gifford 2003). The eight roles of the MTR-I model are listed in Appendix C.

4.3.2 Mapping Belbin Model to MBTI and Keirsey Theories (Gifford 2003)

Mapping models that are created based on roles along with other theories that are built based on personality types is beneficial for assembling effective teams, especially those that need to solve complex or software development problems. The research of Stevens and Henry (Stevens 1998) tried to establish a relationship between the Belbin roles and the Keirsey temperaments. However, this study was performed on a very small sample size. Look at Appendix D for more information related to this mapping.

4.3.3 Mapping MTR-I to MBTI and Keirsey Theories (Gifford 2003)

The MTR-I model is established based on the MBTI theory: every single role is associated with a combination of two MBTI personality types, forming a total of eight team roles. The author of the MTR-I justified that any personality type could be fitted properly into any team role. The Curator is mapped to the personality types of ISTJ, and ISFJ. The Innovator is mapped to the personality types of INFJ, and INTJ. The Scientist is mapped to the personality types of ISTP, and INTP. The Crusader is mapped to the personality types of ISFP, and INFP. The Sculptor is mapped to the personality types of ESTP, and ESFP. The Explorer is mapped to the personality types of ENFP, and ENTP. The Coach is mapped to the personality types of ESFJ, and ENFJ. The Conductor is mapped to the personality types of ESTJ, and ENTJ.

4.3.4 Mapping MTR-I to Belbin Role Theory (Gifford 2003)

The major variation between the MTR-I model and the Belbin role theory is that the MTR-I roles are based on theoretical work while the Belbin roles are based on experiments. Both of these evaluations are quite similar to each other since they measure characteristics and behaviors of each individual and create a profile that includes a detailed description of his/her attitude. Gifford states that “Belbin based his roles on tests measuring inelegance, dominance, and the scales of extraversion-introversion, and stability-anxiety.” However, intelligence is not directly evaluated using the MBTI theory, since intelligence is not a personality characteristic. On the other hand, the Thinking-Feeling dimension does evaluate for the objective and logical; Gifford found that the emphasis in programming is based on the objective and logical rather than the personal aspects.

The original mapping of the MTR-I team roles to the MBTI personality types are integrated with the Belbin roles. However, the role of Chairman is not included since there was a conflict resulting from the lack of sample points. Furthermore, the Company Worker could not theoretically be estimated successfully from the MBTI, but Lyons (Lyons 1985) observed that 22.6% of the computing population is ISTJ. The roles of CW and ME are mapped to the personality types of ISTJ. The role of CW is also mapped to the personality types of ISFJ. The role of PL is mapped to the personality types of INFJ, and INTJ. The role of SH is mapped to the personality types of ESTJ. For more information related to this mapping, look at Appendix E.

4.3.5 Cognitive Team Roles

Cognitive Team Roles is a psychometric model (See Appendix F) that assists project managers in building successful teams based on creating profiles that determine individuals’ cognitive strengths and weakness, preferences, interests, and flexibilities. The psychological base that is used for building this model is “Thinking Styles” from which ten roles are proposed, based on people’s social behaviors and task focus. The Cognitive Team Roles theory increases the performance of team members and improves teamwork by reducing stresses and giving them the confidence to take responsibilities, perform to their best abilities and deal effectively with critical circumstances. Additionally, this model focuses on cognitive roles and the socio-dynamics of the teams more than the functional roles. In other words, when an individual’s profile is located, the thinking interests of that team member are attached to it, not his/her functional roles (Beddoes-Jones 2001; Beddoes-Jones 2002).

Chapter 5

Cultural Aspects

5.1 Overview

Project managers from many disciplines find themselves dealing with aspects of international management, interacting with foreign affiliates, traveling to different countries, and negotiating with various team members who grew up in different environments and cultures. With the global culture of software products, this is even more the case for software project managers. Sherriton asserts "Culture affects many aspects of our personal and professional life" (Sherriton 1997) such as the way we interact with people in our work situations and in the international settings in general. Therefore, software project managers should be mindful the international cultural environment and its influence on the team members' roles. In general, each country has its own political and economic agenda, technological status and development level, regulatory environment, comparative and competitive advantages, and cultural norms and values. Thus, the task of software project managers is becoming more difficult as they are responsible not only for assembling teams based on their skills, experience, and knowledge, but also to analyze the new environment that consists of people from different cultures. Like managers of other disciplines, software project managers must be able to develop suitable strategies and procedures to effectively plan, organize, lead, monitor, and control individuals according to their local regulations and expectations, skills and capabilities, social interaction, norms and values, and ethical behaviors (Deresky 1997).

Many researchers have explored factors that consistently lead to effective cross-cultural management. They observed that there are some differences and similarities between the various cultures' managerial skills that led the teams to succeed, but they have not determined a general set of management behaviors. Further, they noticed that every country or every culture has its own common factors, personalities, values, and behaviors that contribute to producing high quality results. Therefore, researchers currently focus on developing theories of social behavior in the working environments of various cultures in order to enable managers practically and effectively deal with other team members' cultures (Deresky 1997).

This chapter examines culture as a relevant aspect of managing software projects, provides a conceptual framework that helps software project managers to reason about related cultural variables, and outlines cultural profiles for team members from different countries.

5.2 The Role of Culture in Team Management

Understanding the nature, dimensions, and variables of a particular culture helps project managers recognize the perspective of team members who come from different societies. Several studies indicate that team members' lack of cultural awareness costs organizations money and requires extra management effort (Clark 1990; Simon 1990). Studying the effects of cultural variables and dimensions on teamwork enables managers to create suitable policies and procedures, identify valuable plans, and organize, lead, and control a specific team based on a particular cultural setting. Software project managers should understand and appreciate cultural diversity in their efforts to assemble useful teams. In general, teams which involve people from various cultures increase the probability of having large or small differences in the behavior and attitude of members, since every culture has its own communication styles, values, standards, and expectations (Clark 1990).

Table 5.1 summarizes some key results from national, societal, and cultural variables, as well as other issues such as attitudes and individual behavior. Both the national and societal variables provide the development framework and the perpetuation of cultural variables. These cultural variables identify fundamental attitudes toward work, time, materialism, individualism, and change. The attitudes, in turn, influence a team member's motivation, expectations, commitments, and ethics with respect to their teamwork relationships. This could finally affect the possible outcomes from every team member (Deresky 1997).

Table 5.1: Environmental Variables Affecting Team Management

National variables	Economic System, Legal System, Political System, Physical Situation, Technological, Know-how
Societal Variables	Religion, Education, Language
Cultural Variables	Values, Norms, Beliefs
Attitudes	Work, Time, Materialism, Individualism, Change
Individual and Group Employee Job Behavior	Motivation, Productivity, Commitment, Ethics

The impact of culture on particular management functions is clearly observed when comparing U.S. values and systems with other societies' norms and values. American managers make plans and schedules, develop activities, and meet their deadlines based on people's beliefs and abilities that have the greatest impact on the future. However, managers from Islamic nations might believe that events could occur based not only on successful planning, but also on the will of God. Another study shows that a number of Arab oil workers who got their training in Texas, observed that the American teaching style is impersonal, something opposite to their teaching method which is easily influenced by personal feelings (Harris 1991).

Cultures also show the potential to adapt to various constraints or situations that may confront their culture. For example, many Japanese employees at a U.S. manufacturing plant learned how to communicate and interrupt conversations when there was a serious problem, something different from Japanese culture, since they were not used to arguing while other people were still talking (Harris 1991).

5.2.1 Cultural Variables and Values

With the world's various cultures and subcultures, software project managers should understand the particular nature of a certain people by building a cultural profile for each region or country of the members represented on the team. To support this, there are eight universal cultural variables identified and categorized based on the overall character of a specific group. The first category is "Kinship," which is defined as the system that is adopted by a certain society to direct family relationships. For example, the Kinship system in the United States is represented by single-parent families or the nuclear family, contrary to Eastern nations, since their system consists of an extend families with many individuals from various generations. The second aspect is “Education” which defines the level of education, training programs, or the degree that an individual should gain in order to be accepted and recruited in an organization for a particular country (Harris 1991).

The third category is “Economy” which describes the effect of economic system on people in a particular society in terms of production and distribution. The fourth concept is “Politics” which explains the degree of flexibility and freedom from government, and the degree to which each government imposes constraints and regulations on an organization (Harris 1991).

The fifth category is “Religion,” which defines the spiritual beliefs of a society and the base that underlies the moral and economic norms. For example, the impact of religion in the workplace in the U.S. is limited since the belief is largely in hard work and a strong work ethic. However, Hindus and Muslims believe in the concept of destiny; Muslims in particular always invoke the phrase “God willing” in making their decisions. In Western countries, some religious organizations such as the Roman Catholic Church play a major cultural role in terms of moral and political issues (Harris 1991).

The sixth aspect is “Associations,” which describes the societies that come up with their own system of rules based on religious, social, professional, and trade relationships. The seventh category is “Health,” which identifies the role of health in increasing the organizations’ productivity and performance by taking care of its individuals (Harris 1991).

The last aspect is “Recreation” which describes the way that people in an organization use their leisure time, and how this could affect their behaviors and attitudes towards their jobs (Deresky 1997). For example, in the U.S. culture, it is largely thought that 2 weeks vacation is acceptable. However, in most European cultures there is an expectation of a much longer “holiday.

Values specify how people would appropriately respond in any given situation, based on their societies’ ideas about what is good or bad, and what is right or wrong. Deresky proposes four value dimensions: power distance, uncertainty avoidance, individualism, and masculinity. The *power distance* is defined as the level of acceptance by any society of distributing non-identical power in organizations. For example some countries display a high level of power distance, such as Malaysia and Mexico. Others, such as Austria and Israel, display low power distance; in these, most individuals have equal power, possibly resulting in more harmony and collaboration (Hofstede 1983; Ronen 1985).

The *uncertainty avoidance* describes the level where people in a society feel threatened by unclear circumstances. For example, Japan and Greece have strict rules and procedures that provide more security and career stability. Managers in those countries tend not to take high risks; the employees are more patient and have a strong sense of nationalism. On the other hand, countries such as United States, United Kingdom, and Denmark prefer to follow less structured or less formal activities and are willing to take more risks (Hofstede 1983; Ronen 1985).

Individualism refers to the tendency of people only to look at themselves and their families without paying attention to their society's needs. For example, people in Singapore and Korea have strong social frameworks as well as an emotional sense of belonging to the organization. However, democracy, along with individual programs, plans, and achievements, are highly encouraged in the Unites States and Australia, for example (Hofstede 1983; Ronen 1985).

The term *masculinity* describes "the degree of traditionally "masculine" values: assertiveness, materialism, and a lack of concern for others that prevail in a society." For example, Japan and Arab countries are highly masculine societies; women are mainly responsible for staying home and raising a family. However, women in Swaziland participate in high-level jobs. The United States lies in the middle since American women are encouraged to work as well as take care of their children (Hofstede 1983; Ronen 1985).

Nath and Sadhu (Nath 1988) provide a summary of cultural dimensions based on geographical region. North Americans score high in individualism, low in power distance, medium in uncertainty avoidance, and high on masculine. Japanese score high in collectivism, medium in power distance, high in uncertainty avoidance, and equally in masculine and feminine. Europeans score equally in individualism and collectivism, medium in power distance and uncertainty avoidance, and equally in masculine and feminine. Chinese score high in collectivism, low in power distance and uncertainty avoidance, and equally in masculine and feminine. Africans score high in collectivism, high in power distance and uncertainty avoidance, and high in feminine. Latin Americans score high in collectivism, high in power distance and uncertainty avoidance, and high in masculine. However, there is some sort of flexibility in Europe; some countries have high power distance and uncertainty avoidance such as Latin Europe while others have low power distance and uncertainty avoidance such as Anglo Europe.

5.2.2 Critical Operational Value Differences

Operational value differences have a substantial effect on an individual's attitudes and behaviors, and indicate the way that people react to various circumstances. Time is one of the most important values in the world, especially for Americans since they are used to saving, scheduling, and spending it with a high level of accuracy. Americans believe that time is money: deadlines and schedules must be met successfully. However, people in Arab countries or in Latin America always translate the time concept as an indefinite time in the near future: for example, they are not more specific than saying "tomorrow," for example. Arabs believe that important things must take a lot of time to accomplish, therefore, they only rush on tasks that do not need a lot of work or respect (Deresky 1997).

The second essential value is the change concept: Western countries generally believe that a person can get some control over the future and might be able to manipulate events, while Arab regions believe in destiny and that God is the only one who has external control over the future. Chinese people respect their traditions; changing their lives or at least modifying some of their concepts is not easy at all. Furthermore, Americans use natural resources more than any other people in the world. For example, Indians and Koreans consider nature to be part of their religious beliefs. In general, Americans prefer to work independently, appreciating individual accomplishments, promotions, and wealth rather than any group objectives. However, the Chinese enjoy creating a social life and a collaborative work environment (Deresky 1997).

5.2.3 Cultural Profiles

In this subsection, profiles of American, Japanese, German, and Korean cultures are presented to understand general cultural values and variables of North American, European, and Asian. These are included here because they most likely present the majority of people working in software industries. However, cultural profiles of other regions and countries in particular are also important in software development and will be discussed later in this chapter such as the profiles of Arab countries, Middle Eastern, Indian, Chinese, Latin American, etc.

5.2.3.1 America

Americans think that they can achieve anything with enough time, technology, and money. They enjoy working in an organization that is organized and secure, with everything in the right place. Americans fought a revolution and wars to keep their concept of democracy; they therefore feel annoyed or angry at undesirable control from their government. Generally, they believe in that all people are created equal; however, not everyone is able to accept that ideal. They also believe that they can create the future they desire with the help of their aspirations and motivations. Furthermore, Americans have a strong work ethic, spend their time effectively, accomplish tasks, and always search for new technologies and methodologies. Americans do not like the traditional privileges of royalty and class stratification. They are informal in dress and greetings, but they usually avoid embracing in public. In addition, Americans are oriented in teamwork and a social environment since they look to successfully accomplish tasks. Traditional American values consist of "family loyalty, respect for all ages, marriage and the nuclear family, patriotism, material acquisition, forthrightness, and the like." Moreover, Americans like to share with people, help refugees, participate in aid programs, and assist neighbors when needed (Deresky 1997).

5.2.3.2 Japan

The major principles of Japanese life are peace and harmony. They gain their values from the Shinto religion, which concentrates on spiritual and physical harmony. They believe that confidence, faith, and honor are essential factors for building effective relationships. Japan displays a high level of power distance, uncertainty avoidance, masculinity, and pragmatism as well as loyalty and empathy. They prefer to work in an organization that is much similar to a family system including a strong manager, powerful working environment, a seniority system that ranks people according to their services, and a system that regularly monitors individuals. The Japanese are willing to cooperate and share information with their groups. They enjoy looking for solutions, solving problems, making critical decisions, and are able to create a long-term vision. The Japanese grow up learning how to resolve conflicts in order to achieve tasks, and to avoid the shame of not accomplishing their duties. Generally, Japanese rank high on collectivism and feel comfortable with the hierarchy system (Deresky 1997).

5.2.3.3 Germany

Germany displays a high level of uncertainty avoidance and masculinity, but has a low level of power distance. Germans rank high on individualism; however, their behaviors and attitudes appear to be less individualistic than those of Americans. Therefore, Germans enjoy being around familiar people in different circumstances. They seem to love rules, and respect what is allowed and what is prohibited, perhaps because many of them are Catholics or Protestants. Furthermore, Germans know how to make best use of their time to complete their work, but also to have fun. They tend to show a great deal of confidence in their work environment, but without aggression. They follow the hierarchy system and make decisions after getting official approval from the department head. They also prefer the closed-door policy, more personal space during conversations, and privacy in aural distance. In general, Germans are conservative, appreciating privacy, politeness, and formality. Moreover, Germans require detailed information before and during discussions in order to effectively negotiate. They make sure that their speech is controlled; not wordy but hitting the main point, since Germany is considered as a low-context society where communication is limited (Gannon 1994).

5.2.3.4 Korea

Koreans score high on collectivism, pragmatism, and uncertainty avoidance, but rank low on masculinity and average on power distance. Koreans focus on their traditional teaching styles, which rely on the concepts of spiritualism and collectivism. Moreover, they respect family authority, formality, and class differences. They are hard workers, friendly, very hospitable, and demonstrative, but also quite aggressive. They believe that family and personal relationships are important. Therefore, they enjoy establishing strong relationships with others and care about gaining social and professional reputations. On the other hand, they like to work cooperatively with other colleagues but do not prefer to participate in management (Deresky 1997).

5.2.3.5 Key Observations from Profiles

The key observation from the previous profiles is that people are different and they have different attitudes and features since they belong to different cultures. In particular, they have various perspectives regarding technology, time, and money. Some of them use their times accomplishing tasks and always look for new technologies. Others enjoy communicating with people to explain problems and resolve conflicts. On the other hand, some people do not prefer to participate in management while others strongly go for it.

It is obvious that software managers should understand the cultural profiles in order to recruit the right person for the right mission. For example, a software team leader should be interested in managing and coordinating people in order to successfully handle his/her job. Also, a requirements engineer should be able to communicate effectively with the customer to successfully extract his/her needs. Thus, cultural profiles would help software project managers improve their selection task.

5.3 The Cross-Cultural Communication Environment

Communication is one of the most important factors in cross-cultural management, since it is the core of motivation, leadership, group interactions discussions, and negotiation. Therefore, software project managers should understand the relationship between culture and communication in order to be able to effectively write, talk, and listen across cultural boundaries. A software project manager's capability in communicating successfully with various cultures would contribute to producing high quality products. In general, communication explains the process by which people share meanings, by exchanging messages via media using words, behavior, attitude, or even physical objects. Software project managers communicate and interact as a way of coordinating activities, distributing information and ideas, motivating individuals, as well as negotiating future actions. The communication process in software projects can be quite complex, since people understand messages and interpret them according to their own expectations and perceptions of reality as well as their values, norms, and attitudes (Sahay 2003).

Deresky discusses seven cultural variables that might independently influence the communication process: attitudes, social organization, thought patterns, roles, language (spoken or written), nonverbal communication, and time. They are discussed here to give an appreciation for the complexities that can arise in software projects.

1. Attitudes: This variable determines the way people think, feel, behave and interpret messages from others. The problem occurs when an individual assumes that every single person in the society has similar features, behaviors, and traits. This could lead to a critical confusion and create a big misunderstanding in cross-cultural communication. This also could cause a big conflict and critical delay when the requirements engineer proposes the customer needs in way that could be understood differently by the software architect and designer.
2. Social Organization: This variable describes the way that values, methods, and priorities of a particular social organization could affect people's behaviors. Generally, these organizations are established according to a specific nation, government, community, tribe, or religious group.

3. Thought Patterns: This variable determines the influence of variations in the logical progression of reasoning between the different cultures. In other words, people have different reasoning processes and software project managers should be aware of that.
4. Roles: This variable specifies the different ways in which people recognize the role of managers. In other words, some cultures assume that managers are the only ones who make decisions and assign responsibilities. Therefore, software project managers should be aware of that since the cultural profile of Japanese, for example, shows that they prefer to follow their family system that has a strong manager in which, this could delay the process of making their design decisions.
5. Language: This variable is one of the most important factors that could affect the communication process, not only because it might cause problems for people who belong to various cultures, but also for individuals who belong to the same culture or country. The problem occurs when someone have difficulty expressing his/her feelings, is unable to speak the local language fluently, is unable to translate idioms, and misunderstands specific body language or symbol. For example, more than 800 languages are spoken in Africa; each one has its own structure and terminologies. Moreover, more than 14 official and many unofficial languages are used in India. Thus, software managers should understand this critical issue since most software project teams consists of people from the entire world.
6. Nonverbal communication (body language): This variable describes the behavior that causes interaction and communication but without words. People prefer to see more than hear since "a picture is worth a thousand words." Nonverbal communication can be classified into four categories as follow:
    a. Kinesics behavior: Body movements such as postures, gestures, facial expressions, and eye contact.
    b. Proxemics: Describes the impact of the nearness level and space on communication (both personal and work space).
    c. Paralanguage: "Refers to how something is said rather than the content – the rate of speech, the tone and inflection of voice, other noises, laughing, or yawning."

   d. Object language: Also called material culture, determines how people communicate through material artifacts.

7. Time: This variable identifies the different way that people treat and use time. For example, time for Middle Easterners is controlled by the will of God. However, time for Americans and Germans means a lot and they deal with it as something to be spent accurately, saved, or even wasted properly. They experience time linearly with a past, present, and future. This provides an indication that Americans and Germans can handle the job of a software project manager in order to deliver the final product on time.

Cultures have different levels of contexts: some cultures are considered as high-context cultures such as Asian, Middle Eastern, African, and Mediterranean cultures, while others considered as low-context cultures such as German, Swiss, and North American cultures. The problem occurs and a conflict arises when high context people communicate with low context people, because feelings and ideas are not clearly expressed by high context people, which makes it more complicated for low context people to understand and interpret. In contrast, low context people express their feelings and ideas in a few meaningful words that directly hit the main point. In general, the following countries are ordered from high to low context: Japan, Middle East, Latin America, Africa, Mediterranean, England, France, North America, Scandinavia, Germany, and Switzerland (Deresky 1997). In general, a requirements engineer who belongs to a high context culture may produce detailed requirements specification document that makes it clear and easy for the designer to understand. However, a software project manager who belongs to a low context culture would explain that main objective of the work directly without confusing the team members and give them the opportunity to effectively brainstorm.

5.3.1 Communication in South Korea

The communication process in South Korea is deeply affected by cultural characteristics. “Saving face” is one of the most important aspects of South Koreans’ communication;

they believe that interactions result from building a good image of themselves. Therefore, they are usually aware of losing their good reputation in order to avoid harming their relationships. They also make sure not to upset an individual's feelings or bother his/her mood. Koreans prefer the hierarchy system in which people communicate and are coordinated according to their rank and class. However, each individual should be respected in both communications and interpersonal relationships. In other words, they follow a protocol that requires people to politely use rituals of introductions, bowing, and praise when meeting each other. Generally, the communication method is South Korea is structured as well as "vertical, top-down, and highly implicit." South Koreans have the ability to read someone's face and have enough experience to react to both verbal and nonverbal signals. Therefore, foreign software project team members should be careful while communicating with Koreans because they may incorrectly interpret nonverbal cues, potentially destroying the relationship. Korean culture teaches people how to be social, guess what others are thinking, listen carefully to conversations, and understand them using eye contact. This means that Koreans are patient and could handle the job of a requirements engineer while communicating with the customers to understand their needs carefully. On the other hand, Koreans prefer to be silent, listening more than talking. Moreover, they speak loudly when they want to point out something important. They also feel much more comfortable using a very small amount of personal space in their daily communications and interactions (Suh 1972).

5.3.2 Communication in Arab Countries

Arabs are frequently enthusiastic, warm, sensitive, and emotional. The Arabic language consists of many expressive words, exaggerated terms, and a lot of adjectives. However, "What is said is often not as important as how it is said, regardless of the content." For example, Arabs tend to speak loudly in dramatic situations or when they criticize some one. In general, Middle Eastern culture depends strongly on religion, honor, loyalty, friendship, and traditional hospitality. Social relationships between families and friends have a strong impact on business issues. Arabs create business with people, not organizations; they make commitments to individuals, not to contracts. They often solve critical problems with a simple phone call to a close friend or a family member. They often believe that relationships never end, and they understand that give and take is important in growing strong relationships (Axtell 1985).

Arabs are hospitable; it's a part of their culture to provide refreshments and beverages to others, talk about social life and suddenly rush into work discussions without giving any introduction. They tend to be slightly introverted until a relationship is completely built; that might take a long time. Arabs often use the expression of "tomorrow if God wills" when preparing plans and procedures. In addition, Arabs start their communications with small talk, discuss business for a moment, and then chat about general issues before getting to the main point of the talk (Axtell 1985). Therefore, Arabs could handle the job of requirements engineer and software architect and designer since they enjoy communicating with others and know how to attract people into conversations and that helps understanding the software conflicts. Furthermore, they present their ideas in details that could help the software designers understand the entire system easily.

5.3.3 Communication in Japan and United States

Communication between Japanese and American people is a good case in point for examining communication in software project teams. These two cultures differ significantly in their style of communicating. For example, Japanese use indirect verbal and nonverbal communication while Americans are more direct. Japanese make decision in private while Americans frequently make it in public. Japanese negotiators make their decisions in along term while Americans prefer to negotiate in a short term. Moreover, Japanese combine business with social communication while Americans tend to separate the business from the social aspects (Goldman 1994).

The differences between Japanese and American communication styles underline a key reason why a model is needed that incorporates culture into the team configuration process for software project managers. In other words, software projects today consists of many people from the entire world in which, understanding their communication styles would help software managers resolve conflicts and manage risks quickly to avoid delays and deliver the final product on time and within budget. For example, a software project manager should understand that Japanese have good ideas and decisions but may not propose it in public in which, he/she should expect that these valuable thoughts will be delivered soon in private or via email messages.

5.4 The Cross-Cultural Negotiation

Negotiation is an important process that is considered as a middleware between planning and implementation. It describes the process of discussion between two or more individuals as a purpose of reaching a mutually acceptable agreement. Thus, the capability to conduct a useful and successful negotiation with businesspeople all over the world would result in keeping business, preserving relationships with alliances, avoiding confusion and delays, and making effective agreements and decisions. Therefore, software project managers should have the ability to understand the various cultures of practitioners in order to successfully negotiate solid agreements and plans. Cultural differences cause critical difficulties in the negotiation process since every country or culture has its own way of negotiation. Essential variations between cultures in the negotiation process involve the quantity and kind of preparation for a negotiation, the comparative emphasis on responsibilities versus interpersonal relationships, the dependence on general principles and standards rather than particular aspects, and the number of parties and their effect on the negotiation process. This means that good managers should be familiar with various cultural backgrounds and their negotiation tactics and procedures in order to effectively control, make progress on, and achieve an organization's objectives. In general, recognizing the differences between various cultures' negotiation styles is not an easy task, since every culture has its own values, lifestyles, expectations, methods to formal procedures, and techniques for solving problems (Harris 1991).

5.4.1 Negotiation Styles

This subsection discusses the different negotiation styles of several countries to understand the impact on software project managers who organize software project teams with heterogeneous cultures involved. The negotiation process consists of five phases: preparation, relationship building, the exchange of task-related information, persuasion, and concessions and agreements. Thus, software project managers should understand the cultural variations in negotiating styles by comparing countries' cultural profiles. The profile includes the value system of a particular country, attitudes, norms, and behaviors (Deresky 1997).

5.4.1.1 America

American negotiators tend to know how to bargain, stand on a solid base when they begin negotiations, do not make concessions before something logically happens, and "keep their cards close to their chest." Moreover, Americans tend not to refuse compromises when the end of their negotiations results in deadlock. They also set up general standards and hand over the detail work to associates, and try to keep many options open to ensure a flexible negotiation. Americans tend to work in good faith, respect the other side, try to present their opinions clearly, and know how to control the negotiation process in an effective manner. Furthermore, Americans explain their approaches briefly and directly, taking the value of time into consideration, and try to hide their position as long as possible to let the other side reveal their position. Additionally, they tend to make sure to allow other negotiators to move forward first in order to gain a comparative advantage (Deresky 1997).

5.4.1.2 India

Indian negotiators tend to be tenacious, patient, and persistent; they often follow Gandhi's principles, which "combine strength with the love of truth." In other words, they try to negotiate with opponents based on right reasons. Indians tend not to be afraid of speaking loudly, have the ability to control themselves, and try to find solutions that make everybody happy. Moreover, they respect other negotiators by being patient and not hurting their feelings; they avoid using violence or insults. In addition, they are flexible, changing their minds easily, but they take unpredictable risks more seriously. They can also effectively move their attention from small details to the larger picture. They are humble, trust other negotiators, use silence, and are able to withdraw at any time. Furthermore, Indians tend to be self-sufficient, demonstrate their inner resources and strengths, and appeal to the other side's spiritual identity. They do not keep secrets, try to learn from other negotiators, and prefer to trust their ideas and beliefs more than logical reasoning (Deresky 1997).

5.4.1.3 Arabs

Arab negotiators tend to follow Islamic customs: they often show respect, honor, and dignity to the other side. Because of these characteristics, Arabs are often respected and trusted by people from all nations. They avoid direct disagreements between negotiators, do not let them show weakness or admit defeat in critical situations, and use their prestige to encourage others to listen to them carefully. Moreover, Arabs are creative and endeavor to find a general solution that all parties can agree on, appreciate the different parties' circumstances, and have the ability to resist many kinds of stress or pressure during the negotiation process. They also persuade other negotiators to change their minds using references to people who are highly respected, they are able to keep secrets, and can usually control their anger and emotion. Additionally, Arab negotiators tend to use conferences as a mediating diplomacy, are able to deal with the Arab tendency to disregard time, and recognize the influence of Islam on the other negotiators who believe that they hold the truth, follow the right path, and will win based on justice (Deresky 1997).

5.4.1.4 Sweden

Swedish negotiators tend to be quiet, remarkably polite, punctual, and have thoughtful ideas and opinions. They are straightforward, and show a strong desire or willingness to produce high quality products. Swedish negotiators also tend to have the ability to control their feelings, emotions, and anger in sensitive situations. However, they can be perceived as conceited, and perfectionist—they are satisfied with no less than perfection.

Additionally, they are very private and try to avoid face-to-face confrontations (Deresky 1997).

5.4.1.5 Italy

Italian negotiators tend to have "a sense of drama in which acting is the main part of the culture." They are often direct and do not conceal their emotions, know how to read facial expressions and gestures, and tend not to trust easily. Moreover, they tend to make decisions based on the first impressions of the other negotiators and are often have a sense of history. Italian negotiators often prefer working individually more than in teams, are constantly willing to help, but seldom hold certain suggestions. Furthermore, they often know how to come up with new strategies and solutions, may be critical of other negotiators' ideas, and can include others in complex negotiations. Italians often "handle confrontations of power with subtlety and tact, flair of intrigue, and know how to use flattery (Deresky 1997)."

5.4.1.6 China

The Chinese are "the toughest negotiators in the world (Deresky 1997)"; they put much greater weight on respect, friendship, and reputation than Americans. They often require more details about product features in order to decide whether they accept or refuse a deal. Chinese negotiators spend a lot of time coming to an agreement. The typical negotiation style in China has little authority, opposite to Americans who have authority and want to conduct a deal. The main objectives of Chinese negotiators are often treated within the framework of state planning and political standards. The negotiation process in China is influenced by three cultural values: "their ingrained politeness and emotional restraint, their emphasis on social obligations, and their belief in the interconnection of work, family, and friendship (Deresky 1997)." Therefore, negotiation or persuasion in China can be quite complicated and difficult. Chinese negotiators tend to withdraw from a negotiation in order to avoid opening conflicts. They feel much comfortable negotiating with familiar and trusted individuals, and make deals easily with people with whom they have strong relationships. Yet, they seldom make final agreements until negotiations are finished (Deresky 1997).

5.4.1.7 Differences between North American, Japanese, and Latin America

As we know, software project teams involve people from different cultures who have various negotiation styles. Therefore, understanding these styles would help software managers control the project and mitigate risks effectively. Generally, Japanese hide their emotions, not argumentative, and sometimes affected by their social relations and other interests. Moreover, Japanese prefer accurate and valid results to make their decisions. North Americans deal straightforwardly with situations, make decisions based on a cost benefit basis, argumentative, and good at resolving conflicts while negotiating. Latin Americans are more social in which they believe that what is good for a group is good for an individual, argumentative, but has a strong loyalty to a family (Casse 1982).

Based on the information above and from the software management perspective, you can see that Japanese should have a valid and accurate requirements specification document in order to negotiate and make their design decisions. However, their design decisions could be influenced by their personal interests in which, a project manager should be aware of that. North Americans are direct and deal straightforward with people in which, they can handle the job of a team leader negotiating directly with individuals without wasting time hiding their emotions. Generally, every culture has its own positive and negative aspects which require a software project manager understand how to make the best use of every positive feature.

5.4.1.8 Differences between North Americans, Arabs, and Russians

North Americans rely on logic and objective facts, good at establishing relationships, and respect deadlines. Arabs are emotional, follow their subjective feelings, but establish social relationships in long terms. Russians assert ideals, make small concessions, but ignore deadlines (Glenn 1984).

Understanding the different negotiation styles between cultures would help software managers control the project and handle risks. North American can effectively handle the job of a project manager who can deliver the final software product on time. However, Russians ignore deadlines which might strongly delay the project and increase the risks.

5.5 Cross-Cultural Decision Making

The manner and the speed in which decisions are made have a substantial influence on the negotiation process. Decision-making is a part of the managers' and team leaders' everyday routine; decision-making needs to be performed properly, since this activity could require a lot of time and effort if not managed properly. Therefore, managers should understand the impact of cultural differences on decision-making processes and approaches (Fisher 1980).

The decision-making process consists of the following phases: definition of the problem, collection and analysis of data, determining alternative solutions, deciding the best solution, and making the right decision. What varies the cultural impact from one country to another is whether a particular culture follows an objective approach or a subjective approach. Western managers interpret a situation and propose alternative solutions based on objective information. However, Latin Americans are more subjective; they make decisions based on their emotions and feelings (Fisher 1980).

Another important cultural variable that could affect the decision-making procedure is the risk tolerance of individuals making the decision. Generally, American managers have the highest tolerance for risk. Many studies illustrate that individuals from Japan and Netherlands have a higher tolerance for risk than individuals from Germany, Belgium, and Austria (England 1978; Fisher 1980). Another essential variable in the decision-making procedure is the manager's awareness of the locus of control over outcomes. In other words, some managers believe that they can control and make effective decisions that could change or affect the future while others believe that controlling the future is not reachable and is only in the hands of God, fate, or nature. Some American managers believe robustly in self-determination; they identify problematic circumstances in a way that they can control and make useful changes while others do not. On the other hand, managers from Indonesia and Malaysia always feel that they do not have enough capability to influence the future (England 1978).

One more thing that could affect the deliberation of alternative solutions, considered as an important variable, is managers' feelings towards choosing well-known solutions or trying to find new ones. European managers make decisions based on previous experiences and emphasize quality. However, Americans "are more future oriented" and always try to explore new ideas and solutions (England 1978).

"The relative level of utilitarianism versus moral idealism in any society affects its overall approach to problem (Deresky 1997)." In general, utilitarianism strongly influences decision-making behavior in Western countries. Canadian managers are significantly more utilitarian than Chinese leaders, who move toward problems from a viewpoint of moral idealism. More specifically, the Chinese think about the problem, possible alternatives and solutions, from a long-term and social perspective rather than from a personal perspective (Deresky 1997).

Another important issue that could also affect the process of decision-making is that of autocratic versus participative leadership. In other words, this tendency determines the people who have permission and authorization to make decisions. For example, Germany, Turkey, China, and India follow the hierarchical system in which the authorization for action has to be passed upward to the department head or top management before making final decisions. However, the system of authorization in Sweden is decentralized. Americans are perhaps in the middle between autocratic and participative leadership styles. Arab managers make their decisions based on the customs and traditions of their religions. Generally, Middle Eastern people seem to handle business in an extremely personalized way; top managers or leaders control final decisions. The Japanese, however, give emphasis to collective harmony, group decision-making, and general agreements (Deresky 1997).

The final aspect of the decision-making process is how fast or slow decisions are made. North Americans and Europeans make quick decisions since they usually tend to have definite results. However, Middle Easterners make decisions slowly, since they spend more time on critical or important issues. The Japanese spend extensive time in the early stages of the decision-making process, and then identify whether there is in fact a real need for a decision or not (Deresky 1997).

5.6 Cross-Cultural Leadership

The core of leadership is to aid employees to recognize their highest capability at work. Thus, the major objective of every leader is to encourage individuals to produce to their best ability and to satisfy the organization's requirements. Today, international managers should be flexible enough to learn and change their previous culturally conditioned styles of leadership and to look at the future (Robinson 1984). Software project management requires a project manager and a team leader who can deal with different people from various cultures in order to encourage them to produce to their best abilities.

Successful leaders are those who have the ability to motivate and encourage people to have creative thoughts, attitudes, and behavior. However, poor or weak leaders could possibly prevent an organization from meeting its objectives. The multicultural leader attempts to increase leadership value by managing and organizing essential, and sometimes conflicting, roles as: "a representative of the parent firm, the manager of the local firm, a resident of the local community, a citizen of either the host country or of another country, a member of profession, and a member of a family (Robinson 1984)." The role of a leader involves the interaction between two sets of variables: the content and the context of leadership. The content of leadership consists of the leader's attributes and traits, and the decisions to be taken, while the context of leadership includes all variables that are relevant to a specific organization. In particular, the content of leadership involves knowledge of job position, experience, expectations, decisions and personal work style, personality, values, beliefs, ability to change, as well as intelligence and cultural learning. However, the context of leadership involves resource availability, physical location, and technical requirements of the job (Robinson 1984).

5.6.1 Leadership Around the World

Previous studies demonstrate that individuals in countries that score high on power distance, such as India, Mexico, and the Philippines, prefer an autocratic leadership mode and some paternalism. This is because they feel more comfortable with an obvious distinction between leaders and those in lower positions, rather than confusion over who has decision-making responsibility. However, individuals in countries that score low on power distance, such as Sweden and Israel, prefer a consultative, participative, leadership mode in which they assume superiors to handle that style. Researchers ranked the following countries from lowest to highest based on the presence of autocratic norms: Germany, France, Belgium, Japan, Italy, the United Sates, the Netherlands, Britain, and India (Mason 1987). Another study shows that Americans and Germans allow individuals to participate more in decision-making than Italians and Japanese, while Indonesians stick strongly with the autocratic leadership style (Deresky 1997). Another study illustrates that leaders in Sweden, the Netherlands, the United States, Denmark, and Great Britain encourage team members to participate in problem-solving while leaders in Italy, Indonesia, and Japan do not give chances to its individuals to propose and discuss their approaches and solutions (Mason 1987).

Indians prefer to work better individually rather than in groups, since management in India is usually autocratic, based on a hierarchical system with strict authority and charisma. Moreover, the decision-making process is centralized, forcing people to focus more on rules and procedures and to avoid taking high risks. However, Indian culture highlights moral orientation and loyalty, teaching individuals to make relationships and work hard for the society, not only for personal goals (Deresky 1997).

Japanese management style is effective and successful because people in Japan follow the process of continuous improvement; they work on small tasks every day until they are finished. The main purpose of that is to mitigate errors and defects, avoid risks, and control and ensure quality. Leadership in Japan is highly participative; they ask for advice and suggestions, encouraging their fellows to share ideas, exchange knowledge, and make decisions and consultations. They also prefer to work in groups to allocate solutions to problems in a collaborative manner. Additionally, the Japanese seek to achieve the organization's goals more than their personal aims (Deresky 1997).

5.6.1.1 Differences between Middle Eastern and Western

Software project managers should understand the different management practices between various cultures to successfully organize, evaluate, control, monitor, lead, communicate, and make decisions. Deresky compares Middle Eastern and Western management practices based on organizational design, patterns of decision making, performance evaluation and control, manpower policies, leadership, communication, and management methods (Deresky 1997). Middle Easterners are highly bureaucratic, do planning randomly, unwillingness to take risks, and follow informal control strategies. Additionally, they depend strongly on social relations to solve problems, and sometimes affected by their family power. They also follow old and outdated management principles. On the other hand, Westerners are less bureaucratic, use modern tools and more scientific methods, as well as focus on cost reduction and quality of the products.

The above information indicates that there is an obvious difference between Middle Eastern and Western practices in which, software project managers should be aware of while selecting their team members. In other words, software project managers should recognize how different cultures communicate, negotiate, react to the different situations, do planning, and make decisions.

This chapter examined culture as a relevant aspect of managing software projects, provided a conceptual framework that helps software project managers to reason about related cultural variables, and outlined cultural profiles for team members from different countries. This chapter also provided a key contribution to the configuration model that will be discussed in the next chapter.

Chapter 6

Discussion

6.1 The RPC Model Framework

This chapter discusses how the Belbin model coupled with the Myers-Briggs model and Keirsey theories could be applied effectively to software development teams based on previous studies. In particular, this research focuses on five roles in software engineering: project manager, team leader, requirements engineer, architect, and designer. The study not only explores the relationship between Belbin roles and Myers-Briggs personality types but also examines how cultural differences, with respect to key values and variables, could be mapped successfully into the personality profiles and ultimately the team configuration model. As a result of this research investigation, a model is presented that extends current work on software project team configuration to incorporate cultural aspects of team members into the configuration decision. This model is intended to help software project managers reason about key factors that influence software project team configurations for software projects based on (RPC) roles, personality, and cultural profiles, as shown in Figure 6.1.

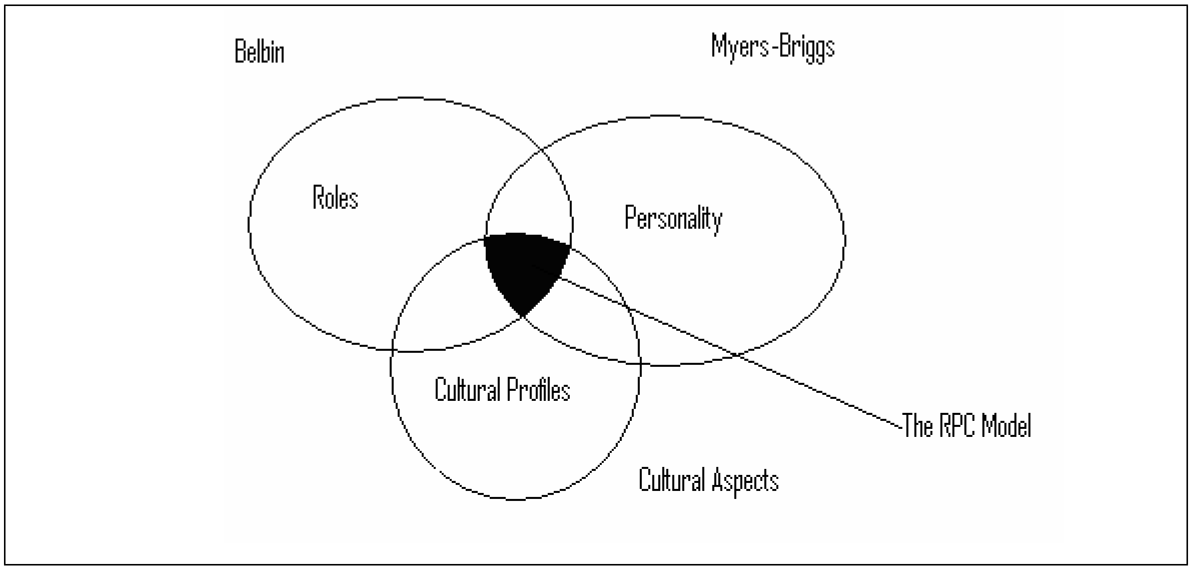

Figure 6.1: The RPC Model Framework

6.2 Mapping RPC into the Role of Software Project Manager

The major tasks of a software project manager are to plan (including the estimation of cost, time, and effort); to monitor and control people, processes, risks, and events as the software evolves from a preliminary concept to an operational deployment. The project manager is also responsible for identifying the problems and tasks to be conducted, obtaining the required resources, and specifying the work products, constraints, and goals. The project manager creates quality assurance checkpoints and mechanisms to mitigate risks and control unexpected changes. In addition, the project manager is responsible for organizing and motivating team members to produce to their best abilities. The project manager coordinates the interface between business specialists and software professionals. A software project manager does all this to deliver high quality software, on time and within budget.

According to Belbin's studies, the team member who scores high in Chairman (CH), Completer-Finisher (CF), and Team Worker (TW) roles could best fit the responsibility of a project manager. However, the highest priority is for the one who scores higher in the CH and CF roles. In other words, scoring high in the TW role is useful but not essential based on Stevens' studies (Stevens 1998). As we discussed in chapter 4, Gifford's experiment (Gifford 2003) showed that the CF role has a substantial effect on the project's progress since around 83% of the software teams with Completer-Finishers completed their tasks successfully. Stevens' studies (Stevens 1998) demonstrated that the leadership role of the CH has significant effect on software development teams. Stevens (Stevens 1998) also proved that the role of the TW is beneficial to the entire team since the TW encourages individuals to work as group, as well as ensures that everything is under control.

This process ensures that the project manager is a self-confident person who can control the team, has the ability to listen to other team members' opinions and ideas, and makes progress towards accomplishing both of the group objectives and organization goals. The project manager should demonstrate a very strong desire to achieve perfection, maintains the sense of necessity within the entire team to meet deadlines and goals, and delivers products on time and within budget. Additionally, the project manager supports individuals in their strengths and weakness, and socially communicates and negotiates with all team members to find optimal solutions.

Based on the Myers-Briggs and Keirsey Theories' mappings and arguments, the best personality type for a project manager is an ESTJ, ENTJ, ESFJ, or ENFJ, depending on the nature of the task. As we discussed in chapter 4, Gorla (Gorla 2004) demonstrated that software team leaders with intuitive (N) personality type perform better than those with sensing (S) type in information gathering. Moreover, Gorla discovered that software managers with a feeling (F) type perform better than those with thinking (T) type in making decisions. Gorla also showed that software managers with a judging (J) personality type would ensure delivering the final product on time and within budget. Peslak's research (Peslak 2006) proved that extroverted, judging, and thinking people all together improve the success of the project. Bardley's study (Bradley 1997) showed that the best personality type for a project manager or a leader is an ESTJ, or ENTJ. Kroeger (Kroeger 1992) proved that the best team leader personality is an ESFJ.

These personality characteristics ensure that the project manager communicates effectively with team members and get important things via social activities, prefers a planned and structured life, and manages individuals to fulfill goals. However, a project manager with the (N) personality type is preferred in a situation that requires using imagination to understand something without conscious reasoning or study, while a project manager with the (S) type is needed for a situation that requires making decisions based on facts, details, and procedures. A project manager with the (T) personality type is required when information should be organized in a logical and objective manner. In contrast, an open-minded project manager with the (F) type is needed when information should be categorized in personal way.

The cultural profiles discussed in Chapter 5 show that Western managers, including Americans and Europeans in general, but not Russians, would handle the role of a project manager. Americans are often good at planning and scheduling, believe in their capabilities to change and improve the environment, and are more realistic in their aspirations. They work hard to achieve their goals and make the best use of their time. In other words, Americans save, schedule, and spend time with high a level of accuracy. Moreover, they have strong work ethics and appreciate individuals' accomplishments and promotions. They often follow scientific methods and search for the most recent technologies and methodologies. Americans are often more direct in verbal and nonverbal communication, tend to be more committed to a task, give more immediate feedback, and make decisions in public at the negotiating table. Furthermore, Americans know how to bargain, keep many options open to ensure flexibility, as well as explain their approaches directly, briefly, and clearly. They tend to make decisions based on definite results. In general, it seems that Americans have the ability to accomplish tasks with enough time, technology, and money.

European managers, or more specifically Germans, enjoy being around people, and respect and are willing to follow rules. Moreover, Germans know how to make best use of their time to achieve their tasks successfully, and tend to show a great deal of confidence when in their work environment. They control their speech: they are not wordy but speak directly. Swedish managers also respect the value of time and control their feelings, emotions, and angers in sensitive situations. They always try to produce a perfect and unique piece of work. In general, European managers make decisions based on previous experiences and tend to emphasize quality.

6.3 Mapping RPC into the Role of Software Team Leader

The major tasks of a software team leader are to organize teams in an appropriate way that maximizes each member's ability, knowledge, and experience, and to motivate and encourage individuals to come up with innovative ideas. The team leader is responsible for pushing the entire team to produce to their best abilities, facilitate communication and collaboration, and to improving team members' arguments in decision-making and in taking action. The team leader should also understand individuals' emotional needs, show respect, kindness, and appreciation, and insist on solving problems and getting the job done effectively. The team leader is the one who builds commitment and confidence creating a sense of destiny and purpose among all team members. Additionally, the team leader always asks members to work hard, and focuses on high performance goals and metrics.

The role of a software team leader is quite similar to the role of a project manager. Thus, the same Belbin roles, Myers-Briggs personality types, and cultural aspects apply to the role of a software team leader, except one role. In other words, the team leader who scores high in Shaper (SH), Completer-Finisher (CF), and Team Worker (TW) roles, is best to take on the responsibility of a software team leader; the highest priority is for the one who scores higher in the SH and CF roles. The Chairman (CH) can still handle the job but the Shaper (SH) is preferable (Stevens 1998). In general, the Shaper allows team members to focus their attention on the core of objectives and priorities, and enforces some shape or pattern on group discussion and on the result of group activities. Generally, a shaper has a managerial style that is different than the leadership of a Chairman. A shaper encourages individuals to work to their best ability and try to find all possible solutions to the different problems. However, shapers are competitive and argumentative.

6.4 Mapping RPC into the Role of Software Requirements Engineer

The major tasks of a software requirements engineer are to identify functional and non-functional requirements, as well as provide both the architect and the designer a complete and accurate requirements specification document. The requirements engineer is responsible for conducting meetings with the customer, understanding his/her needs, negotiating limited business issues and conflicting requirements, and determining how the end users will interact with the final product. More specifically, the requirements engineer should be able to negotiate a logical approach, suggest unambiguous solutions, as well as verify and validate the specification. In general, the software requirements engineer specifies the user scenarios, builds up a refined technical model of software functions and features, identifies classes and services, produces UML diagrams, and constructs a bridge to design and implementation.

Based on Belbin studies and arguments, the requirements engineer who scores high in the Resource Investigator (RI), Company Worker (CW), and Monitor-Evaluator (ME) roles would best fit to take the responsibility of a project manager. However, the highest priority is for one who scores higher in the RI and CW roles. In other words, scoring high in the ME role is useful but not necessary, since prior research provides some evidence that the role of ME might be beneficial (Stevens 1998; Gifford 2003). As we discussed in Chapter 4, Stevens (Stevens 1998) illustrated that the CW role has a significant effect on software development project, translating the project plans and concepts into practical procedures and methods. Stevens showed that the RI role also has a substantial effect on software development progress, exploring new things and useful ideas.

This process ensures that the requirements engineer is able to communicate effectively with people, find out what they need, and discover creative ideas and approaches. Moreover, the requirements engineer can turn concepts and plans into practical working procedures and rules, and organize the specifications in a systematic manner. The requirements engineer also can analyze the problems easily and evaluate suggestions, methods, and alternatives.

According to Myers-Briggs and Keirsey Theories' mappings and arguments, the best personality type for a requirements engineer is an ESTJ, ESFJ, ISTJ, or ISFJ. As we discussed in Chapter 4, Gorla (Gorla 2004) demonstrated that a requirements engineer with a thinking (T) personality type perform better than those with feeling (F) type. Gorla also showed that software engineer with a judging (J) personality type would effectively deal with the external world. Peslak (Peslak 2006) proved that extroverted, judging, and thinking people all together improve the success of the project. Bradley (Bradley 1997) discovered that extrovert, sensing, and thinking engineers would increase the level of communication between the stakeholders. Moreover, Gifford (Gifford 2003) mapped the roles of CW and ME into the personality types of ISFJ and ISTJ, respectively.

These personality characteristics ensure that the requirements engineer prefers a planned and structured life and can effectively make decisions based on their senses as well as on facts, details, and procedures. However, a requirements engineer with the (E) personality type is preferred in a situation that requires the requirements engineer to communicate socially with the customer to determine his/her needs, while a requirements engineer with the (I) type is recommended in a situation that does not require a high level of interaction, to avoid interruptions and confusions. A requirements engineer with the (T) personality type is required when information should be organized in a logical and objective manner. In contrast, an open-minded requirements engineer with the (F) type is needed when information should be categorized in a personal way.

The cultural profiles as described in Chapter 5 indicate that Asians, including Japanese and Koreans, as well as Middle Easterners, would handle the job of a requirements engineer. The Japanese build relationships based on confidence, faith, and honor, and enjoy cooperating and sharing information with their teammates. They have the ability to find solutions, solve problems, make critical decisions, and create a long-term vision. Furthermore, the Japanese demonstrate preferences to graphics more than to text (Sahay 2003). Koreans are social, hard workers, and friendly; able to listen to conversations, intuit what is on others' minds, and understand what they want. Middle Easterners are friendly and have a unique communication style that helps other stakeholders express their feelings and thoughts in a relaxed way. They focus on achieving the required tasks, not only to meet deadlines, but also to create strong relationships with all parties. Middle Eastern people are also creative and able to find a general solution that all stakeholders can agree on; they also show respect and appreciation to others' opinions.

On the other hand, Indians and Chinese can also handle the role of a requirement engineer, but may be less preferable. Indians are social, patient, demonstrate their resources and strengths, and have the ability to control themselves in critical circumstances. They focus on details and explain requirements specifications clearly. However, Indians prefer to trust their ideas and beliefs more than logical reasoning, and sometimes prefer to work alone. The Chinese prefer to follow plans and procedures, focus on details, but are considered as the toughest negotiators in the world since they spend extra time on the negotiation process.

6.5 Mapping RPC into the Role of Software Architect

The major tasks of a software architect are to establish a blueprint that provides an overall picture of the system including data structure, software components, architectural styles, and the interrelationships between the different components. The architect is responsible for identifying alternative architectural styles or patterns to generate the structure that best satisfies the customer's requirements and the quality attributes. Furthermore, the architect has to elaborate the design to get useful details regarding the entire architecture, and to review the final architectural design to ensure its correctness, clarity, completeness, and consistency. Generally, the most important task of a software architect is to highlight early design decisions that will have a significant effect on the next phases of the development.

According to Belbin's studies and arguments, the software architect who scores highest in Plant (PL), Company Worker (CW), and Monitor-Evaluator (ME) roles could be the best fit to take the responsibility of a software designer. However, the highest priority is for the one who scores higher in the PL and CW roles. In other words, scoring high in the ME role is useful but not necessary, since prior research provides some evidence that the role of ME might be beneficial (Stevens 1998; Gifford 2003), as mentioned previously in this chapter. As we discussed in Chapter 4, Stevens (Stevens 1998) illustrated that the CW role has a significant effect on software development project, translating the project plans and concepts into practical procedures and methods. Stevens also showed that teams with PL individuals had better performance since they always look for creative and innovative ideas and solutions.

This process ensures that the software architect is able to brainstorm and come up with innovative ideas, approaches, and solutions. Moreover, the architect can turn concepts and plans into practical working procedures and rules, and organize the specifications in a systematic manner. The architect can also analyze the problems easily and evaluate suggestions, methods, and alternatives.

Based on Myers-Briggs and Keirsey Theories' mappings and arguments, the best personality type for a software architect is an INTJ, ISTJ, INFJ, or ISFJ. Gifford (Gifford 2003) mapped the roles of CW, ME, and PL into the personality types of ISFJ, ISTJ, and INTJ respectively. These personality characteristics ensure that the architect utilizes of plans and structures and makes best use of his/her personal thoughts and feelings. An architect with the (N) personality type is preferred in a situation that requires using imagination to understand something without conscious reasoning or study, while an architect with the (S) type is needed for a situation that requires making decisions based on their senses as much as on facts, details, and procedures. An architect with the (T) personality type is required when information should be organized in a logical and objective way. In contrast, an open-minded software architect with the (F) type is needed when information should be categorized in personal manner.

The cultural profiles as described in Chapter 5 indicate that Americans, Europeans, Chinese, Middle Eastern, Japanese, and Indians would handle the job of a software architect. Western people in general, interpret a situation and propose alternative solutions based on objective information. Americans often define a problem, determine alternative solutions, and decide on the best solution. They also tend to handle risks and make appropriate design decisions. They believe that they can make effective decisions that could change and improve the entire system. Europeans make design decisions based on previous experience, emphasize quality, and try to find new solutions and approaches. The Chinese tend to think about the problem, alternatives, and solutions from a long-term and social aspect rather than a personal aspect. On the other hand, Middle Easterners, the Japanese, and Indians tend to work hard and provide emphasis on collective harmony, group decision-making and general agreements.

6.6 Mapping RPC into the Role of Software Designer

The major tasks of a software designer are to set up standards, procedures, concepts, and practices, as well as to integrate customer requirements, business needs, and technical aspects in order to help developers build the system. More specifically, the software designer is responsible for evaluating the information domain model, designing a suitable data structure for data objects and attributes, and selecting an architectural style and pattern from the analysis model. Moreover, the designer often divides the analysis model into design subsystems, develops a set of design classes or components, designs an interface for external systems, and designs user interfaces. Generally, the most important tasks of a software designer are to build a deployment model and produce an unambiguous design specification document.

The role of a software designer is similar to the role of an architect. Thus, same Belbin roles, Myers-Briggs personality types, and cultural aspects are applied to the role of a designer.

6.7 The RPC Model

The RPC model is intended to support the software project manager in reasoning about decisions for software team configurations. This model is built based on previous studies' results and conclusions of Belbin theory and Myers-Briggs model. Additionally, the proposed model maps the cultural profiles into the software engineering roles based on prior cultural studies. The RPC model is summarized in Table 6.1.

Table 6.1: The RPC Model

	Roles	Personality	Cultural Profiles
Software Project Manager	CH CF TW *	ESTJ ENTJ ESFJ ENFJ	Americans Europeans
Software Team Leader	SH CF TW *	ESTJ ENTJ ESFJ ENFJ	Americans Europeans
Software Requirements Engineer	RI CW ME *	ESTJ ESFJ ISTJ ISFJ	Japanese Koreans Middle Eastern Indians ** Chinese **
Software Architect & Designer	PL CW ME *	INTJ INFJ ISTJ ISFJ	Americans Europeans Chinese Middle Eastern Indians Japanese

* Means: Not necessary, ** means: Less priority.

This model is built based on previous studies of Belbin roles and Myers-Briggs personality types. Those studies provide some evidence that Belbin roles and Myer-Briggs personality types are effective for software development projects in which, further studies are needed to be conducted in order to evaluate and validate the RPC model. Moreover, the cultural aspects are presented and mapped into this model based on prior studies in international management in which, further investigations are needed to verify and validate the cultural values and variables within the software engineering context. Thus, this work creates a good basis for further study and research in software project management.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Contributions from this research include:

1. A framework to serve as the basis for examining cultural issues in the context of software project teams.
2. A preliminary software team configuration model that demonstrates how a software project manager would reason about software team configurations – aligning roles with skills/knowledge/experience, personalities, and culture.
3. An initial research corpse of knowledge from which to explore mappings between personalities, roles, and culture.

In other words, this research proposed a model for identifying and selecting effective staff for software project team configurations based on roles, personality, and cultural values and variables; a project manager could therefore reduce risks and handle the conflicts of software development efforts. Generally, this thesis discussed how the Belbin model, as well as the Myers-Briggs model and Keirsey theories, and certain cultural aspects could be applied effectively to software development teams based on previous studies and arguments.

More specifically, this research studied the existing team configuration models that have been developed for general teambuilding, such as the Belbin model and the Myers-Briggs and Keirsey theories, and mapped which are applicable or appropriate to software engineering tasks and roles based on prior software studies and arguments. Particular attention was given to the roles of a project manager, team leader, requirements engineer, architect, and designer. In addition, this study not only investigated how those models could be applied to software development teams but also explored the relationship between personality profiles and cultural differences.

This study found evidence that all Belbin roles are important and could be applied successfully to software projects, except the Monitor-Evaluator role, since prior research provides some evidence that the role of ME might be beneficial to software teams but is not necessary. Furthermore, every software engineering role requires a different set of Belbin roles depending on the nature of the task. The four personality perspectives of the Myers-Briggs Type Indictor are applied successfully to software team configuration, all except the Perceiving personality type. Moreover, every software engineering role requires a different combination of MBTI personality types depending on the nature of the task.

Understanding the cultural profiles of all individuals aids project managers in selecting appropriate team members to handle various roles. For example, Americans and Europeans tend to work best as project managers and leaders, according to the RPC model. However, this does not mean that only Americans and Europeans can handle the job successfully, but at least provides some indication that may increase the probability of recruiting the most effective team member for a certain mission.

This research has some limitations since it focuses on just four cultural profiles: North American, European, Asian, and Middle Eastern. Although this thesis discusses some cultural perspectives of Africans, Latin Americans, and Australians, the available information is not sufficient to end up with a solid conclusion. Furthermore, the RPC model focuses only on the management roles as well as the roles of requirements engineer, architect, and designer. On the other hand, there is some validation beyond opinionative reasoning about the connections between roles, personalities, and culture since we did not have enough time to run experiments to significantly verify and validate the RPC model. However, this piece of work is considered as the precursor to a more comprehensive research effort in my Ph.D program.

7.2 Future Work

This research is by no means conclusive. There is a great deal more work that must be done before the RPC model can be effectively conveyed in the project manager's team configuration decision process. Thus, this research is not only a Masters thesis but also a good basis to start working on my Ph.D proposal. Here is what we will be doing next:

1. Evaluate the RPC model by conducting studies and running experiments to verify and validate the model.
    a. Study the effect of the RPC model on the overall software project success.
    b. Study the effect of the PRC model on the overall software team processes.
    c. Study the effect of the PRC model on the overall software team performance and productivity.
2. Extend this research and explore the cultural perspectives of Africans, Latin and South Americans, as well as Australians.
3. Expand this study and involve other software development roles such as the role of programmer, tester, system engineer, quality assurance engineer, etc.

References

Axtell, R., and ed. (1985). Dos and Taboos Around the World New York John Wiley and Sons.

Bailey, J., and Stefaniak, G. (2001). "Abilities needed by computer programmers." SIGCPR - ACM: 93-99.

Beaver, J., and Schiavone, G. (2006). "The effects of development team skill on software product quality." ACM SIGSOFT **31**(3): 1-5.

Beddoes-Jones, F. (2001). "Think Smart Report." from www.cognitivefitness.co.uk.

Beddoes-Jones, F. (2002, Sep, 28, 2006). "Belbin Team Roles and Cognitive Team Roles: A study of two perspectives?" from http://www.cognitivefitness.co.uk/thinking_styles/articles/UK%20HRD%202003.pdf#search=%22belbin%20team%20roles%20and%20cognitive%20team%20roles%22.

Belbin, M. (1981, 1996). Management Teams. New York, USA, John Wiley and Sons.

Biddle, B. (1979). Role Theory: Expectations, Identifies, and Behaviors. New York, USA.

Bradley, J., and Hebert, F. (1997). "The effect of personality type on team performance " Journal of Management Development **16**(5): 337-353.

Budd, J. (1992). 15FQ Technical Manual
Letchworth: Psytech International Ltd.

Canning, M., Tuchinsky, M., and Campbell, C. (2005). Building Effective Teams Duke, USA, DUKE CORPORATE EDUCATION, Dearborn Trade Publishing.

Casse, P. (1982). Training for the Multicultural Manager: A Practical and Cross-Cultural Approach to the Management of People Washington, D.C, Society for Intercultural Education

Chung, W. (1994). "Effects of participative management on the performance of software development teams " SIGCPR - ACM **3**(94): 252-260.

Clark, A. (1990). "Japan Goes to Europe." World Monitor: 36-40.

Constantine, L. (1993). "Work organization: Paradigmes for project managment and organization " CACM **36**(10): 34 - 43.

Costa, J., and McCrae, R. (1992). Four ways five factors are basic, Personality and Individual Differences

Cushing, J., Cunningham, K., and Freeman, G. (2003) Towards Best Practices in Software Teamwork. The Evergreen State College, Seattle, WA - IDX Corporation, Olympia, WA **Volume**, DOI:

DeMarco, T. (1987). Peopleware: Productive Projects and Teams. New York, USA, Dorset House Publishing Co.

Deresky, H. (1997). International Management: Managing Across Borders and Cultures. United States of America, Addison-Wesley Educational Publishers Inc.

Edgemon, J. (1995). "Right stuff: how to recognize it when selecting a project manager." Application Development Trends **2**(5): 37-42.

England, G. (1978). "Managers and Their Value Systems: A Five-Country Comparative Study " Journal of World Business **13**(2).

Faraj, S., and Sproull, L. (2000). "Coordination Expertise in Sofwtare Development Teams " Management Science **46**(12): 1554-1568.

Fisher, G. (1980). International Negotiation: A Cross-Cultural Perspective. Chicago, Intercultural Press.

Galton, B. (2003) Team Building. Defence Aviation Repair Agency (DARA) **Volume**, DOI:

Gannon, M. (1994). Understanding Global Cultures Thousands Oaks, CA: Sage Publications

Garlan, D., Jackson, M., Mead, N., Potts, C., Reubenstein, H., and Shekaran, M. (1994). "The role of software architecture in requirements engineeing " IEEE: 239 - 245

Gifford, S. (2003). A Roadmap for a Successful Software Development Team Assembly Model Using Roles. Computer Science and Applications. Blacksburg, Virginia Tech. **Master of Science**.

Glenn, E., Witmeyer, D., and Stevenson, K. (1984). "Cultural Styles of Persuation " International Journal of Intercultural Relations.

Goldman, A. (1994). "The Centrality of Nigensei to Japanese Negotiating and Interpersonal Relationships: Implications for U.S. Japanese Communuication " International Journal of Intercultural Relations **18**(1).

Gorla, N., and Lam, Y. (2004). "Who Should Work with Whom? Builfing Effective Software Project Teams." Communication of the ACM **47**(6): 79-82.

Hackman, R. (1990). Groups That Work and Those That Don't. San Francisco, USA, Jossy-Bass Publishers.

Harris, C. (2003). Building innovative teams, Palgrave macmillan.

Harris, P., and Moran, R. (1991). Managing Cultural Differences Gulf Publishing Company.

Henry, S., and Stevens, K. (1998). "Using Belbin's Leadership Role to Improve team Effectiveness: An Emprical Invitigation." Journal of Systems and Software **44**.

Higgs, M. (1996). A comparison of the Myers-Briggs Type Indicator and Belbin Team Roles, Henley Management College.

Hofstede, G. (1983). "National Cultures in Four Dimensions." International Studies of Management and Organization.

Howard, A. (2001). "Software Engineering Project Management." Communication of the ACM **44**(5): 23-24.

Javed, T., Maqsood, M., and Durrani, Q. (2004). "A Survey to Examine the Effects of Team Communication on Job Satisfication in Software Industry." ACM SIGSOFT **29**(2).

Jiang, J., Motwani, j., and Margulis, T. (1997). "IS Team Projects: IS Professionals rate six criteria for assessing effectiveness." MCP Team Performance Management **3**(4): 1-7.

John, M., Maurer, F., and Tessem, B. (2005). "Human and social factors of software engineering " ICSE - ACM.

Jones, F. (2000) Belbin's team roles and cognitive team roles: A study of two perspectives? **Volume**, DOI:

Katzenbach, J., Jon,. and Smith, D. (1993). The Wisdom of Teams, Harper Bsiness.

Keirsey, B., David, and Marliyn. (1984). Please Understand Me Del Mar, California, Prometheus Nemesis Book Company.

Kraut, R. a. S., L. (1995). "Coordination in software development." Communication of the ACM **38**: 69-81.

Krishna, S., Sahay, S., and Walsham, G. (2004). "Managing Cross-Cultural Issues in Global Software Outsourcing " Communication of the ACM **47**(4): 62-66.

Kroeger, O., and Thuesen, M. (1992). Type Talk at Work. New York, Delacorte Press.

Lawrence, G. (1994). People Types and Triger Stripes. Gainesville, FL, Center for Application of Psycological Types.

Layman, L., Cornwell, T., and Williams, L. (2006). "Perosnality Types, Learning Styles, and an Agile Approach to Software Engineering Education." ACM.

Linda, B., et al. (2002). A Quick Guide to the 16 Types in Organizations: Understanding Persoanlity Differences in the WorkPlace, Telos Publications.

Lyons, a. M., L. (1985). "The DP Psyche." Datamation: 103-110.

Margerison, C., McCann, and Dick. (1990). Team Re-Engineering. Australia, TMS.

Mason, R., and Spich, R. (1987). Management - An International Perspective Homewood: IL: Irwin

Massey, A., Hung, Yu., Montoya-Weiss, M., and Ramesh, V. (2001). "When Culture and Style Aren't About Clothes: Perceptions of Task-Technology "Fit" in Global Virtual Teams." ACM: 207-213.

McKinney, D., and Denton, L. (2005). "Affective Assessment of Team Skills in Agile CS1 Labs: The Good, the Bad, and the Ugly." ACM SIGSOFT: 465-469.

Miller, M. (2002). "The Belbin team role profiles ", from www.sabrehq.com/team-building-examples/belbin_team_role_info.doc

Myers, B., I., and McCaulley, M. (1985). A Guide to the development and Use of the Myers-Briggs Type Indicator. California Consulting Psychologists Press, Inc.

Myers, B., I., McCaulley, M., Quenck, N., and Hammer, A. (1998). MBTI Manual: A Guide to the Development and Use of the Myers-Briggs Type Indicator. Palo Alto, California, Consulting Psychologists Press, Inc. .

Myers. and Briggs, I. (1987). Introduction to Type. California, USA, Consulting Psychologists Press, Inc.

Nash, S., and Redwine, S. (1988). "People and organizations in software production." ACM SIGCPR **11**(3): 10-21.

Nath, R., and Sadhu, K. (1988). Comparative Analysis, Conclusions, and Future Directions Cambridge, In Comparative Management

Ott, a. L., R. (1993). An Introduction to Statistical Methods and Data Analysis. California, Duxbury Press.

Parker, G., Zielinski, D., and McAdams, J. (2000). Rewarding Teams : Lessons From the Trenches San Francisco, California Jossey-Bass Inc. .

Peslak, A. (2006). "The Impact of Personality on Information Technology Team Projects " SIGMIS-CPR - ACM: 273-279.

Pohl, K., Starke, G., and Peters, P. (1995). "Requirements Engineering: Foundation of Software Quality." ACM SIGSOFT 39 - 45.

Poling, T., Woehr, D., Arciniega, L., and Gorman, A. . (2004). "The Impact of Personality and Values Diversity on Team Performance ".

Pressman, R. (2005). Software Engineering - A Practitioner's Approach. New York - USA, Elizabeth A. Jones - McGraw-Hill series in computer science

Robinson, R. (1984). Internationalization of Business Hinsdale, IL: Drysden Press.

Ronen, S., and Shenkar, O. (1985). "Clustering Countries on Attitudinal Dimensions: A Review and Synthesis " Academy of Management Review **10**(3): 435-454.

Sahay, S., Nicholson, B., and Krishna, S. (2003). Global IT Outsourcing: Software Development across Borders. Cambridge, United Kingdom, Cambrdige University Press.

Sarbin, R. (1954). Role Theory. Massachusetts, USA, Addison-Wesley Publishing Company.

Schoenhoff, K. (2001). Belbin's company Worker: The Self-Perception Inventory and Their Application to Software Engineering Teams

Setlock, L., Fussell, S., and Neuwirth, C. (2004). "Taking It Out of Context: Collaborating within and across Cultures in Face-to-Face Settings and via Instant Messaging " CSCW - ACM **6**(3): 604-613.

Sherriton, J., and Stern, L. (1997). Corporate Culture / Team Culture: Removing the hidden barries to team success. New York, USA, Corporate Management Developers.

Simon, D. (1990). After Tiananmen: What is the Future for Foreign Business in China?, California Management Review**:** 106-108.

Stevens, K. (1998). The effects of roles and personality characteristics on software development team effectiveness Computer Science and Applications. Blacksburg, Virginia Tech. **Doctor of Philosophy**.

Suh, N. (1972). "Management and its Environment in Korea." In Management in an INternational Context 201-225.

Thayer, R., and Dorfman, M. (1997). "Software Requirements Engineering." IEEE Computer Society Press.

Thomsett, R. (1990). "Effective Project Teams " American Programmer: 25 - 35.

Walz, D., Elam, J., and Curtis, B. (1993). "Inside a Software Design Team: Knowledge Acquisition, Sharing, and Integration " Communication of the ACM **36**(10): 63-77.

Weigers, K. (2003). "Twenty-One project management success tips."

Weinberg, G. (1986). On becoming a technical leader Dorset House.

White, K. (1984). "MIS Project Teams: An invistigation of cognitive style implications." MIS **8**(2): 85-101.

Wiegers, K. (2005). "Software peoject management."

Yellen, F., Winniford, A., and Sanford, C. (1995). "Extroversion and Introversion in electronically supported meetings " Information Management **28**: 63-74.

Yourdon, E. (1992). Decline and fall of the american programmer. N.J., Yourdon Press.

Appendices

Appendix A

Gifford's Experiment: Team Arrangements and Constraints

Individuals worked together cooperatively on several parts of the project that were integrated later on by a single team member or by a small group of people. However, this research was limited to the following constraints:

1. Each team should consist of exactly three members assembled from one section.
2. Minimize the difference between the maximum and minimum team grade point average (GPA) by ensuring that each team's GPA is the average of its team members' GPA.
3. Minimize the difference between maximum and minimum team course average (CA) by ensuring that each team's CA is the average of the total computer science courses that have been completed successfully by all three members of the team.
4. A member's role is specified by the highest determined role based on Belbin SPI.
5. Each team should be combined with another team from a different section; the new project group or the new pair should satisfy the following conditions:

    a. Minimize the difference between the maximum and minimum pair GPA by ensuring that each pair's GPA is the average of each of the other paired teams' GPA.
    b. Minimize the difference between the maximum and minimum pair course average (CA) by ensuring that each pair's CA is the average of each of the other paired teams' CA.
    c. Each twin (the other team in a pair) is tagged by Task A or B.
    d. Each pair may have different number of leaders (Shapers and Chairmen); some pairs could have only one leader for only one task or for both tasks A and B.

e. Each pair may have a different number of Completer-Finishers (CF); some pairs have no Completer-Finishers, and others may have one CF for only one task either A or B.
f. Each pair may have different number of Team Workers (TW); some pairs have no Team Workers, and others may have one TW for only one task either A or B.
g. Other roles should be balanced on each pair.

The final arrangement of the teams had minimal variation in average team GPA and CA. However, the final setup of the pairs had minimal variation in average pair GPA and CA. On the other hand, additional considerations were taken in order to get accurate results from an experiment that is relatively close to real world software development projects.

1. The project was to be completed within 5 weeks, not done in less than 90 minutes as in prior studies.
2. The project had two teams with a total of six members; the team of three members was co-located but the two teams of three were not co-located. This is actually different than prior studies where all three members were using only one machine.
3. A bulletin board was made available for communication in order to enable twin teams to exchange information and to communicate in distributed format. This kind of communication was not considered essential for prior studies since both the co-location and the time aspects reduced the complexity of the problems.
4. Testing the project components in which this research requires Task A and Task B to build symbiotic components. In other words, each task should execute the other task's program. In contrast, prior studies did not involve components.
5. The project should be scheduled with other students' responsibilities, and must be finished outside the class time within 5 weeks. However, prior studies were conducted during the class time.

6. Students were motivated to work effectively since their participation counted toward the final grade. In contrast, prior studies asked students to participate by just attending the class.

Appendix B

Personality Descriptions

All information is taken from (Peslak 2006) and the following website:
http://www.personalitytype.com

What's your personality type?

Instructions:

This system for understanding people called Personality Type is based on the work of Swiss Psychologist Carl Jung and two American women, Katherine Briggs and Isabel Briggs Myers, creators of The Myers Briggs Type Indicator Instrument® (MBTI)®. The Myers Briggs Type Indicator® and the MBTI® are registered trademarks of Consulting Psychologists Press, Inc.

1. Where is your energy naturally directed?

Extraverts' energy is directed primarily outward, towards people and things outside of themselves. Introverts' energy is primarily directed inward, towards their own thoughts, perceptions, and reactions. Therefore, Extroverts tend to be more naturally active, expressive, social, and interested in many things, whereas Introverts tend to be more reserved, private, cautions, and interested in fewer interactions, but with greater depth and focus.

2. What kind of information do you naturally notice and remember?

Sensors notice the facts, details, and realities of the world around them whereas Intuitives are more interested in connections and relationships between facts as well as the meaning, or possibilities of the information. Sensors tend to be practical and literal people, who trust past experience and often have good common sense. Intuitives tend to be imaginative, theoretical people who trust their hunches and pride themselves on their creativity.

3. How do you decide or come to conclusions?

Thinkers make decisions based primarily on objective and impersonal criteria – what makes the most sense and what is logical. Feelers make decisions based primarily on their personal values and how they feel about the choices. So, Thinkers tend to be cool, analytical, and are convinced by logical reasoning. Feelers tend to be sensitive, empathetic, and are compelled by extracting circumstances and a constant search for harmony.

4. What kind of environment makes you the most comfortable?

Judgers prefer a structured, ordered, and fairly predictable environment, where they can make decisions and have things settled. Perceivers prefer to experience as much of the world as possible, so they like to keep their options open and are most comfortable adapting. So, Judgers tend to be organized and productive while Perceivers tend to be flexible, curious, and nonconforming.

Appendix C

The Eight Roles of the MTR-I Model

1. Explorer: explorers are those people who often try to discover new approaches, methods, plans, and procedures in order to perform tasks and solve problems. They tend to seek hidden potential in individuals. Furthermore, they always look one step ahead of their present circumstances in order to make progress in discovering hidden paths. They also try to modify or change issues related to current situations in order to enhance or explore new ideas.
2. Innovator: Innovators are those people who make best use of their imagination to generate new and various suggestions, thoughts, and perspectives. They are observant of the world around them; they think about what they have observed, but from different points of view. Additionally, they are capable of thinking deeply about unclear problems, creating radical solutions, establishing long-term vision, and suggesting innovative ideas and insights.
3. Coach: Coaches are those people who establish harmony in their organizations by building rapport with individuals, providing a sympathetic understanding and agreement to their difficult situations, ensuring an effective environment for teamwork, and motivating team members. Moreover, they make their best effort to ensure good health, comfortable living, and working conditions for every single member. They also assign different roles to various individuals, encourage them to build strong relationships, ask for opinions and thoughts, and allow them to argue and propose concrete ideas.
4. Crusader: Crusaders are those people who demonstrate interest in specific thoughts and suggestions; they respect values and beliefs. They have the ability to conduct effective discussion sessions, prioritize events, and speak with strong conviction. Moreover, they always seize upon important and valuable thoughts, bringing them to the discussion table and evaluating their potential.
5. Curator: Curators are those people who clarify the inner world of information and thoughts and provide better understanding of different situations. They usually

listen carefully, propose questions, and absorb knowledge by understanding and gaining an obvious picture of every piece of information. In addition, they try to extend their experiences, improve their skills, and look to the future, visualizing clear targets and methods to accomplish their objectives.

6. Sculptor: Sculptors are those people who are experienced enough to simply get things done; achieving what is wanted or hoped for. They have the ability to immediately come up with a sense of urgency in order to accomplish obvious goals and get concrete results. They have the ability to bring individuals together and encourage them to take immediate actions in order to achieve their missions.
7. Scientist: Scientists are those people who introduce structure and method into the inner world of thoughts and understanding. They can explain how and why issues and things occur by stating hypotheses, describing functionality, running experiments, and analyzing and evaluating results. Additionally, they produce logical models that explain the way that a particular aspect performs and try to understand the complexity of any problem or circumstance.
8. Conductor: Conductors are those people who provide method and logical structure to the process by which things are accomplished. They have the ability to construct suitable plans, specify and apply the right procedures, and ensure that the team follows plans and procedures successfully. Furthermore, they assign roles and responsibilities to appropriate individuals according to their skills and capabilities.

Appendix D

More Information on Mapping Belbin Model to MBTI

Generally, there are many advantages for taking into consideration a larger sample size to study and identify the relationship between the Keirsey temperament Sorter and the Belbin Self-Perception Inventory (Schoenhoff 2001):

1. Allocate individuals properly to Belbin roles with respect to their Keirsey personality type.
2. Explore which characteristics of a particular Belbin role are most relevant to the Keirsey personality type. This could only happen if high correlations between some Belbin roles and Keirsey personality dimensions appear.
3. Identify the similarities and differences between the two measurements.
4. Using the Keirsey test to evaluate and form opinions regarding Belbin roles.
5. Discover the breadth of personality scales that best specify software experts, and from that breadth refine Belbin team roles in order to structure effective software teams.

Computer professionals have different types of personalities; in any software organization, there are some variations between designers, analysts, programmers, architects, etc. Lyons discovered that 22.6% of computing experts and only 6% of the general population are ISTJ, while 15.5% of computing experts and only 1% of the general population are INTJ (Lyons 1985). Stevens and Henry (Stevens 1998) observed that there is a variation in the distribution of the Belbin roles. The differences between the distributions indicate that the Belbin role model could be implemented in a different manner to software teams.

In general, the correlations are not statistically significant but at least, to some extent, demonstrate that both the Belbin and MBTI are evaluating same the type of traits and characteristics. The sample size for this study was 102.

It is obvious that the Plant role is an INTJ personality type. In this case, if 15.5% of computing experts are INTJs, and Plants show a strong correlation to that same type of personality, then it is predicted that around 15% of computer experts would be Plants. Furthermore, Schoenhoff's study illustrates that 16.65% of the whole sample where Plants with respect to other roles that are ranked from 3% to 29%. Therefore, it is almost certain that all INTJ individuals are Plants. In other words, identical percentages are expected for both measurements if the INTJ correlates to the Plant. It is clear that a larger sample size is needed to explore more significant trends.

Appendix E

More Information on Mapping MTR-I to Belbin Role Theory (Gifford 2003)

Higgs's report (Higgs 1996) demonstrates the relationships between the MTR-I roles and the Belbin roles, but this study was not conducted on software development projects. Generally, it discusses the variations between the role of Plant, Shaper, Company Worker, etc. Gifford states that the MTR-I model classifies all Plants as Innovators. However, this does not mean that all Innovators should be Plants. This could also apply to other mappings such as the role mapping between Shapers and Conductors, for example. Gifford concludes that this does not mean that the Belbin role theory is better or worse than the MTR-I. On the other hand, all Belbin roles are assigned to exactly two MBTI personality types except the roles of Plant and Shaper that are located only to a single MBTI type. It is also shown that the Company Worker is strongly related to the MTR-I Curator.

Appendix F

The Ten Principles of the Cognitive Team Roles

The Cognitive Team Roles model consists of ten principles; some of them are very much similar to Belbin standards. First, all cognitive roles are essential, but sometimes some roles lead the team to succeed more than others; that depends on the nature of the task and the time within the development life cycle. Second, Cognitive Roles should be taken above the particular functional tasks and responsibilities inherent within the team. Third, there is no specific role that could perform better for the leadership position. The fourth principle states that flexibility is required among all team members when assigned to their role/s. The fifth principle of Cognitive Team Roles model states that some individuals may have multiple roles. The sixth principle requires people who have identical roles to collaborate and work effectively with each other. The seventh principle is that team members should be flexible enough to switch roles over time according to the organization's needs. The eighth principle states that roles should be distributed equally among all team members in order to get a balanced team. The ninth principle states that all cognitive roles need to be fulfilled and satisfied by all team members. The tenth principle is that all roles need to be fulfilled by the all team members to have a successful and productive team (Beddoes-Jones 2002). The Cognitive Team Roles model involves ten roles as listed below (Beddoes-Jones 2002):

1. Intuitive Thinkers: Those kinds of people have strong feelings towards their jobs, tasks, organizations, as well as other team members. They make decisions based on their intuitions, emotions, and sensations.
2. Challengers: Those people challenge an organization's constraints and always try to take some risks just to do what they believe to be right.
3. Altruists: Those people are good at constantly monitoring team members, making best use of their physical and psychological energies to ensure that everything is under control.

4. Collaborators: Those people enjoy building relationships and always try to find chances to work with others.
5. Strategists: Those people seem to think strategically and logically and always have plans and procedures to follow.
6. Creative Thinkers: Those people seem to juggle tasks, look at the problem from different points of view, solve issues backward, and make connections to explore patterns.
7. Logical Thinkers: Those people think in a logical and sequential manner and concentrate on facts, clues, details, and procedures.
8. Detailed Thinkers: Those people focus on both specifics and details, and make sure that all tasks are finished properly.
9. Drivers: Those people enjoy making actions and taking the project forward, progressing from one stage to another.
10. Troubleshooters: Those people enjoy mitigating and managing risks, as well as finding immediate solutions and plans for when things go wrong.

On the other hand, Beddoes-Jones suggests possible correlations between Cognitive Team Roles and Belbin roles. The Challenger could handle the role of the SH. The Collaborator could handle the role of the TW and RI. The Strategist could handle the role of the ME. The Creative Thinker could handle the role of the PL. The Detailed Thinker could handle the role of the CF. Additionally; the Driver could handle the role of the Implementer.