

# Analysis of Japanese Software Business

by

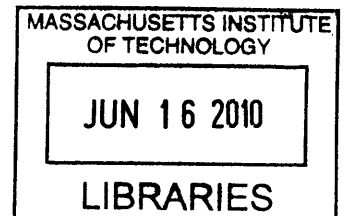
**Kenichiro Inada**

Submitted to the System Design and Management Program  
in Partial Fulfillment of the Requirements for the Degree of

**Master of Science in Engineering Management**

at the  
Massachusetts Institute of Technology  
June 2010

**ARCHIVES**

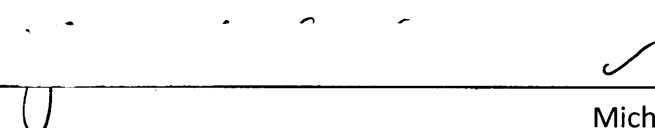


©2010 Massachusetts Institute of Technology. All rights reserved.

Signature of Author \_\_\_\_\_

  
Kenichiro Inada  
System Design and Management Program  
June 2010

Certified by \_\_\_\_\_

  
Michael A. Cusumano  
Thesis Supervisor  
MIT Sloan School of Management

Accepted by \_\_\_\_\_

  
  
Patrick Hale  
Director  
System Design and Management Program



# **Analysis of Japanese Software Business**

by  
Kenichiro Inada

Submitted to the System Design and Management Program  
in Partial Fulfillment of the Requirements for the Degree of

**Master of Science in Engineering Management**

at the  
Massachusetts Institute of Technology

## **Abstract**

Today, our society is surrounded by information system, computers, and software. It is no exaggeration to say that our daily life depends on software and its function. Accordingly, the business of software has made miraculous growth in the last two decades and is playing a significant role in various industries. In accordance with the growing business needs for effective software and information systems, various firms in various countries have entered the business of software seeking for prosperity. Some have succeeded, some have failed. What distinguishes these firms is its ability to manage and deliver quality products on demand, on time, at a low cost. To achieve such goal, software firms have thought out different methods and tools striving to establish its practice. Nevertheless, many software firms around the globe are struggling to satisfy its clients to achieve business success.

With no exception, Japanese software firms are facing difficulties of managing software projects. While its ability to deliver high quality product is well acknowledged among software industry, its high cost structure and schedule delays are thought of as serious problems. Moreover, some of the transitions in the industry are forcing Japanese software firms to seek new opportunities. Therefore, it is important for Japanese software firms to establish more productive ways of developing software products and effective business strategies.

Primal objective of this paper is to analyze the present conditions of Japanese software firms and to derive some recommendations which could enhance its current situation. It will also include the discussion of software development practices in US and India firms to better understand strength and weaknesses of Japanese firms and capture some important concepts which can be applied to improve current practice.

Thesis Supervisor: Michael A. Cusumano

Title: Sloan Management Review Distinguished Professor of Management

## **Acknowledgements**

I would first like to thank my thesis supervisor, Professor Michael A. Cusumano, for supervising my thesis work and providing me warm support and advice throughout. Without his support, this work would not have been as complete nor satisfying. It has been an honor to work with him and will remain as one of my greatest experience of my study here at MIT.

I also would like to thank the SDM community for making my academic experience so fruitful and pleasant. Faculties involved in the SDM program, with abundant knowledge and experience, have inspired me throughout my graduate study in the SDM program. Also staffs at SDM have been extremely supportive and made me feel at home. Lastly, I am thankful to all my classmates of the SDM'09 cohorts who have encouraged me and supported me throughout the program. Their presence has made the experience precious and the friendship we have built will remain as my personal asset for life.

Finally, I would like to thank my wife, Rumi, and my daughter Aoi. Despite the difficulty of adopting in a new environment, Rumi has always been beside me and have supported me. Without her love and assistance, I could not have enjoyed the experience at SDM as much as I did. Aoi who came to Boston when she was 3 months old is now 20 months old. Just watching her grow up has been one of my greatest pleasures. Her presence and warm smile has been the greatest source of my motivation.

# Table of Contents

<b>Chapter 1: Introduction</b> .....	<b>7</b>
1.1 Background .....	7
1.2 Research objective and motivation .....	8
1.3 Overview .....	9
<b>Chapter 2: Key Perspectives</b> .....	<b>12</b>
2.1 Customize software vs. Package Software.....	12
2.2 Onshore development vs. Offshore development .....	15
2.3 Waterfall development vs. Iterative development.....	16
2.4 Traditional application vs. Web application .....	18
<b>Chapter 3: Case Study – Japanese Firm</b> .....	<b>22</b>
3.1 Characteristics of software development at Japanese firms.....	22
3.2 Description of sample company.....	24
3.2.1 Organization.....	25
3.2.2 Company philosophy .....	26
3.2.3 Overview of software development by data .....	29
3.3 Description of sample project.....	34
3.3.1 Overview .....	34
3.3.2 Customer relationship .....	35
3.3.3 Partner relationship.....	37
3.3.4 Offshore development.....	38
3.3.5 Development process .....	40
3.4 Challenges .....	43
<b>Chapter 4: Reference Study – US Firm</b> .....	<b>45</b>
4.1 Basic strategies of software development at US firms .....	46
4.1.1 Hiring policy .....	47

4.1.2 Working in small teams .....	49
4.1.3 How innovation is inspired .....	50
4.2 Agile software development .....	52
4.2.1 Extreme programming .....	54
4.2.1.1 XP Values .....	54
4.2.1.2 XP Practices .....	56
4.2.1.3 How effective is XP? .....	60
4.2.2 SCRUM .....	61
4.3 Capability Maturity Model (Integrated).....	64
<b>Chapter 5: Reference Study – India Firm .....</b>	<b>70</b>
5.1 General advantages of Indian software firms.....	71
5.2 Business strategies of Indian software firms .....	74
5.2.1 Tata Consultancy Service .....	74
5.2.2 Infosys Technologies.....	78
5.2.3 Wipro Technologies .....	81
<b>Chapter 6: Summary and Recommendations .....</b>	<b>86</b>
6.1 Summary of conditions surrounding Japanese software firms .....	86
6.2 Recommendations .....	90
<b>Bibliography.....</b>	<b>94</b>

# Chapter 1: Introduction

---

## **1.1 Background**

As our society became more and more dependent on information systems, which accordingly has become extremely complex especially in the past several years, managing system and software development projects become tremendously challenging in terms of meeting project deadlines and assuring quality of the system. In order to overcome such difficulty, software development firms around the world have thought out and practice numerous methods and tools to improve productivity and quality of software development projects. However, despite all of the efforts, it is true that many of the software development projects today fail to do so. According to the Standish Group's Chaos Report 2009 <sup>1</sup>, only 32% of all projects are delivered on time, on budget, with required features and functions, 44% were either late, over budget, and/or with less than the required features and function and 24% failed which are cancelled prior to completion or delivered and never used. In another words, almost 70% of all software projects are unsuccessful in some way. Among various factors of such failures, most frequently attributed to are incomplete requirements, changing requirements, lack of resources, lack of planning, and unrealistic time frames. Essentially, software development projects are facing difficulties both in managing client relation and internal development activities. As such, this thesis attempts to research and compare different management styles and development practices in different firms in different countries. It aims to investigating what type of development practices are used in different countries and

evaluate its effectiveness. Through investigating different practices, this thesis aims to seek ideal practices and management styles.

## ***1.2 Research objective and motivation***

Primal objective of this thesis is to make suggestions to a Japanese software firm, Company N, in this thesis for anonymity, which specializes in development of customized software. Suggestions will target improvement of current software development practices common to many of the Japanese software firms. Japanese software firms, including company N, are well perceived among the industry as one of the most effective in terms of quality control but with lack of productivity, creativity, and flexibility. To explore the context of such reputation, development practices of company N will be organized to be compared with those of different companies in different parts of the world. As such, this thesis will also include the discussion of some of the practice and management style pursued in software firms of countries beside Japan. Target countries for this investigation will be United States and India which are both important players in the software industry. Investigation will also include the process of understanding societal backgrounds, cultural customs, and corporate strategies which may have resulted in current software development practices. With the result of this comparison, conclusion on whether current Japanese practice can be improved and suggestion on how it can be improved will be presented.



### **1.3 Overview**

Basic structure of this thesis is explained below.

**Chapter 1:** Research objectives and motivation for conducting this research is explained as well as a brief overview.

**Chapter 2:** Key factors which greatly affect how software development projects are managed today are described. Following contrasting factors are presented.

- Customize Software vs. Package Software
- Onshore Development vs. Offshore Development
- Waterfall Development vs. Iterative Development
- Traditional Application vs. Web Application

Insights provided in this section focuses on how different factors affect software development processes and how they are managed. In today's software industry, these four dimensions are considered as a significant determiner. As such, in chapter 2, we will discuss the four dimensions and briefly look into how they affect management of software development projects.

**Chapter 3:** This chapter will primarily be a case study of a Japanese software firm, Company N, based on author's personal experience working at the company. To provide better understanding of a typical Japanese software firm, brief introduction of Company N including its organizational structure, history, and company philosophy will be provided. Also, for the purpose of presenting practical representation of software development style pursued in Japan,

it will discuss activities pursued at a department within Company N which is responsible for managing consultation and system development services for a convenience store chain.

Through the discussions of relationships between various stakeholders of the project, management strategies and development practices will be introduced. Alongside, data extracted from Company N's confidential data will add to the depiction of general characteristics of software development pursued in Japan.

**Chapter 4:** Focus of this chapter is on software firms in United States. First section will be looking at practices of two of the most successful software firms in the world, Google and Microsoft, to understand basic characteristics of firms in US software industry and to introduce some of the business strategies pursued. Then, in the second and third section, we will be looking at “agile software development” and “capability maturity model” in some details, both of which originated from US software development practices.

**Chapter 5:** In chapter 5, we will discuss Indian software industry and how they have evolved into a global player. First, general understanding to what the strength of Indian software firms will be introduced. Then, business strategies and development practices of top three players of India, namely TCS, Infosys, and Wipro, will be introduced. As it will be discussed in the chapter, most software firms of India, including these three firms, share basic strategies in providing software solution services. As such, through the discussion of three firms, this chapter aims to depict strategies of Indian software firms and explore the secrets of its success in the global market.

**Chapter 6:** As a conclusion of this paper, chapter 6 will summarize the findings from three case studies and provide suggestions to software firms of Japan.

# Chapter 2: Key perspectives

---

Before getting into the discussion of characteristics of software firms in different country and its comparison, it is worthwhile to understand what it is that's making these distinctions. In this chapter, some important factors which critically affect software development activities and business operation of individual firms are to be introduced. Such factors, at the same time, are good representations of some important transitions occurring in the software industry. The discussion will present these factors contrastively and briefly explain what each of those factors mean to software development. Some of these factors evolve as a result of firms' occupation or what they aim to achieve, some occur as a result of chosen corporate philosophy and strategy. Through touching on these factors, discussion in this chapter will support the reader in understanding the context of software development activities which will be presented in later chapters. It will also provide some background information of how software firms evolved its current practices and better understand how they can be effective under certain circumstances.

## ***2.1 Customize software vs. Package software***

Type of service or product that a certain software firm specializes in has a significant effect on its software development activities and strategies. In that perspective, there are two types of software, namely customize software and package software. Customize software literally is a custom-made software developed for the purpose of fulfilling business need of a

particular client. In most cases, customize software is only used by a single client with details kept private. Application of customize software, depending on requests of client, extends to entire aspects of business from online shopping site to internal human resource management system. It varies in every aspect of software development including size and function of the software, user of the software, hardware it operates on, and method of delivery. On the other hand, package software in most cases do not assume specific user as a target. Rather, it is meant to provide convenience through the use of software to general consumers or multiple clients who share common business needs. Accordingly, a package software generally includes numerous functions to fulfill as many users' needs as possible. Microsoft's Office series and SAP's ERP products are good examples of package software. Such differences between customize and package software in nature of the service provided and the intended user brings significant difference in development activities pursued at software firms.

First thing that comes up to our mind is the difference in the source of software requirements. In other words, who has the idea of what the software should do. For a customize software, of course, system requirements is with the client where the business needs exist. Accordingly, software firms must conduct series of meetings to understand clients' business needs and what they aim to achieve through the use of intended software. So in a sense, for software firms, requirements definition phase remains somewhat passive. In the case of package software, based on market research and voice of consumers, software firms evolve its own ideas about new solutions to be embedded in a software product which can fulfill consumers' needs. Therein, activities related to defining software requirements are internal. In these regards, defining software requirements brings different set of challenges for software

firms in managing software development. While there is a difficulty of evolving ideas which successfully captures general consumers' needs in package software, there exists significant difficulty of understanding certain client's needs and sharing common image of the intended final product in customize software. Also, in order to understand what the client is looking for, engineers must first understand clients' business and its operation which brings additional challenge for software firms.

Another significant difference is the constraints imposed on software firms in terms of delivery schedule. Most customize software development projects are based on a contract between client and software developer. Important part of such contract is to deliver the software by the intended schedule which often is a critical matter involving important strategic decision on the client side. For example, if it involves client's launch of a new business such as online shopping site, it is critical that the system is delivered and ready to operate on the announced date. Failure to do so will damage client's reputation in the market and as a result, software firms will jeopardize relationship with that client. Therefore, it is absolutely critical that the schedules are met and in most cases they are non-negotiable. On the other hand, in the case of package software, time management is relatively simpler because it does not involve any contracts with any entity. In other words, decision regarding delivery of the product can be made internally. If the delivery of software product is delayed, the only concern is that they will not be making any money until the product is ready. Accordingly, software firms often alter delivery schedule depending on the progression of its development. Nevertheless, there aren't so many complaints because consumers, who choose to purchase certain software prefer

high-quality products with delayed schedule to low-quality products that is delivered on schedule. In that sense, package software has greater liberty in terms of schedule management.

## ***2.2 Onshore development vs. Offshore development***

Offshore development has been a big trend in the software industry over the last decade or so which has brought significant changes in the way software firms manage development projects. The term offshore development indicates the use of labor force outside of the country where particular software firm is located. In contrast, onshore development relies on domestic partners for the manpower to pursue software development projects, particularly for engineers with programming abilities. For software firms, the decision to pursue offshore development evolve around multiple factors such as the nature of software being built, importance of the software, degree of privacy, availability of talents, cost of domestic labor force, and client's location. Of all the factors, cost of domestic labor is by far the greatest determining factor. For many of the IT advanced countries where quality of life is relatively high, cost of domestic labor accordingly has become high. Consequently, in these countries, software firms are striving to secure cheaper labor force through offshore development. In today's highly evolved and globalized environment of the software industry, low-cost delivery has become extremely critical for software firms in order to acquire contracts with clients or to introduce high-quality products to the market with competitive price. In that sense, managing offshore development has become one of the key strategic aspects for software firms.

In return of the significant cost advantage offshore development has brought to software developers, it has imposed great difficulties of managing software development projects. Most of these difficulties arise from communication problems which are caused by differences in culture, language, location, and time zone. In order to handle such difficulties, software firms have thought out different methods and styles in managing software projects which we will be discussing later on in this paper. However, despite all of the efforts, it is also true that engineers involved in offshore development are noticing some inevitable disadvantages in terms of productivity and quality over onshore development. Nevertheless, software firms of the IT advanced countries are striving to benefit from low-cost advantage of offshore development and subsequently have lead to the establishment of numerous downstream developers in the IT emergent countries such as India and China. Subsequently, these two countries has been the primary destination of offshore development for many firms in Europe, United States, and Japan over several years now and have achieved significant growth to acquire major role in the value chain. As it will be discussed in Chapter 5, some of the firms originating from these countries have moved up the value chain to provide upstream value-added service for world's top players in various industries. Such trend is expected to continue and the competition most likely will intensify for existent players. Therein, it is absolutely critical for software firms to appropriately manage offshore development to provide high-value-added services at a low cost.

### ***2.3 Waterfall development vs. Iterative development***



Waterfall model is a traditional software development method where different phases of software development are completed in a sequential order and not recursively. Essentially, output from one phase is an input for the next phase and so forth. Therefore, engineers plan their work based on the premise that all of the tasks from previous phases are completed. Meanwhile, iterative development does not require tasks to be executed in a complete sequential order. Unlike in the waterfall model, tasks are done in parallel wherever possible in the iterative model. So for example, software developers may begin to write codes before completion of the requirement definition phase and ask their clients for feedback to reflect on the work-in-progress system specification documents. Although both models have advantages and disadvantages, the recent trend has been on the iterative development. In today's fast evolving environment where sudden changes in client's business needs are everyday incident, conventional waterfall model does not serve well. To cope with frequent changes in the requirements, many of the software firms are now pursuing iterative development. Extreme programming, which will be discussed in Chapter 4, is a good example of an iterative development.

Despite the significant advantage of pursuing iterative development method, the choice of which development model to pursue is a critical strategic decision for software firms which require consideration of multiple factors such as the two factors discussed earlier in this chapter. First, regarding the type of software a firm provides, waterfall development may fit better with customize software by minimizing the burden on client's side because of limited interactions. In particular, in the case where business needs are clearly defined by the client, waterfall development allows developers to work in a most productive manner. Meanwhile, for

firms which develop package software, iterative development makes more sense because its development activities are concluded internally and recursive execution is not a problem. Moreover, since the process of developing package software often involves try and error type of activity in the attempt to realize abstract idea into a product, iterative development seems to serve better. Secondly, paying attention to the general propensity of pursuing offshore development, it is difficult to determine which of the development style work better. While minimizing interaction by pursuing waterfall development could mitigate the difficulty arising from geographical distance and time zone, limited opportunity to communicate may increase the chance of misunderstanding software requirements. In contrary, iterative development model which requires frequent communication may increase the burden of overcoming communication issues. However, if developers can appropriately overcome such difficulty, iterative development can nullify disadvantage of offshore development and bring out the best result. Overall, there isn't a definite solution on which development style matches certain circumstances. It is important that software firms consider various conditions and all stakeholders to establish own style of development.

## ***2.4 Traditional application vs. Web application***

Last dimension to be considered develops around the type of software or application being developed which also relate to the technologies applied. In a traditional application, which refers to software before the popularization of the internet, main focus was on back office type of application which was limited to internal use and the requirements were

apparent. Moreover, business environment in terms of the rate of change and the degree of new business introduced was much moderate than what we are experiencing today. In such environment, software development was relatively simple and easy to manage. However, as internet usage expanded and web application became popular, complexity of developing software increased tremendously. First, due to the exposure of software application through the internet as represented by an online shopping site, amount of consideration and effort put on user interfaces and usability increased significantly. Consequently, defining software requirements became laborious and involved many changes causing substantial rework. Second, in the internet driven environment where speed of adopting to changes in the business environment is critical, software application which support these businesses are required to stay flexible to change. To handle these complexities accompanied in the web oriented environment, software firms are forced to establish more productive and efficient ways of developing software.

According with the business trend of internet usage and the evolution of web technologies, tools and methods used for software development has evolved. For example, tools such as the CMS (contents management system) and IDE (integrated development environment) are what enable software developers to pursue iterative development while maintaining its productivity. With the evolution of web server technologies along with all of the tools available, software developers are able to reflect and modify application based on feedbacks from a client and compile on web server to get further feedback during a single meeting. Early construction of prototypes and repetitive feedback sessions are good examples of practices which are important components of iterative development method. Such prompt

response was not possible in the era of traditional application. Therein, the evolution of web application and the trend of iterative development may not be discussed separately.

Looking further into the evolution of web technologies, era of the cloud computing which is expected to bring another major shift to the software industry are thought of as a near future. Cloud computing technology, which enables IT solution providers, including software firms, to provide various services through virtual network will further require speed and flexibility on both end of the software business. Type of service that is expected to be provided through cloud computing extends to aspects of hardware, platform, and software which essentially cover every component of software solution service. Consequently, its impact on related stakeholders is outstanding. In the stand point of software providers, ease of distributing software will bring greater opportunity while intensifying competition. In such environment, productivity becomes much critical for software firms. In addition, for those software firms which specializes in customize software, it might be necessary to think flexible and consider package software type of business to make the most of cloud computing. Meanwhile, from the perspective of software users, cloud computing will provide greater options on what type of service to utilize from the cloud, and what to manage on its own as core competence. Moreover, the use of services provided through cloud computing will certainly speed up the software business. For example, in the traditional system development, substantial amount of cost and time is spent on establishing hardware and platform of the system besides developing the software. In cloud computing environment where hardware and platform service can simply be purchased and used via virtual network, main focus of the client will be on delivering innovation and providing attractive contents, essentially the software

portion of a system. Accordingly, expectation on software firms in terms of innovation, technology, cost, quality, and speed will increase dramatically. Overall, coming cloud computing era will require software firms to improve current development practices in various aspects just as we saw when the internet and web application technology popularized.

# Chapter 3: Case Study – Japanese Firm

---

This chapter aims to organize and describe the characteristics of software development practices at Japanese firms. As discussed previously, software development practices are greatly affected by cultural backgrounds and corporate strategy that each firm is based upon. Without exception, many of the characteristics of Japanese software development can be explained as a result of such factors. Therefore, to begin the chapter, some of the characteristics of Japanese software development along with its background and how it has affected the evolution of software development will be introduced.

Second and third section of this chapter will consist of a case study of a Japanese software development firm with focus on particular project which the author has been involved in over seven years. First, general information about the firm and basic philosophy as well as its business strategy will be introduced. Then, focus of the discussion will be shifted to a particular software development project which will describe key activities within software development processes practiced at Company N. Note that name of this organization and sources of data provided are disguised for anonymity.

## ***3.1 Characteristics of software development at Japanese firms***

Professor Cusumano, in his book “The Business of Software<sup>1</sup>” and an article “The Puzzle of Japanese Software<sup>2</sup>”, provides several interesting points about the characteristics of

Japanese software firms and its software development practices. Here are some excerpts from his writings.

- Most systems are custom-built for specific customers in Japan and contain relatively few innovative features, by design.
- The country has underinvested in basic research and higher education. Japan has had relatively weak university training in computers and information systems.
- Major Japanese producers of software have had to train most of their own people on the job and have ended up treating software development mainly as a problem in production.
- Big Japanese companies have tackled large-scale systems development with heuristics, process discipline, some capital, and manpower.
- Good at cranking out multiple versions of custom or semi-custom applications that follow standardized design patterns and evolve little from their original parameters.
- Japanese packaged software development is merely a few percent of the total industry.
- Japanese projects had one-fourth the number of defects reported by customers in the first 12 months after shipment compared to U.S. projects.

All of the points presented above by Professor Cusumano were, for the author, a working software engineer in Japan, very convincing and agreeable. To add on to the points made by Professor Cusumano, below are some of the characteristics which, based on personal

experience, seems to be taking a significant role in terms of shaping software development practices in Japan.

- People of Japan, including software engineers, are culturally and educationally accustomed to following rules or working under strict discipline.
- Japanese industries are traditionally adept in imitating existing products or system to improve them in different ways.
- Labor cost in Japan remains relatively high while investment from various industries on information systems is decreasing.

It seems to be that there are multiple factors responsible for evolving Japanese software development to the state it is now, including its culture, traditions, educational circumstances and economical situations. In the next two sections of this chapter, case study on a Japanese software firm will be presented to discover that these characteristic actually accord with corporate strategies and development methods taken at a typical Japanese software firm. Explanation of some background information and analysis will be provided alongside to support points that were presented in this section.

### ***3.2 Description of a sample company***

This section aims to describe software development at Company N in general and how they are managed. Interestingly enough, characteristics of software development at Company N represents those of typical Japanese firms as discussed in the previous section. In that respect,



this section will provide real live example of how software firms of Japan operates and how it shapes its software development projects.

### **3.2.1 Organization**

Company N is one of the largest IT consulting firm in Japan with over 5,000 employees. The company started its service in the 1960's as a consulting firm as well as a system integrator and has grown steadily over the past several decades. Its broad service covers multiple industries including securities, finance, insurance, distribution, and public services. Historically, Company N has its business strength in the financial and distribution industries. Also, just as most of the large software firms of Japan are, Company N provides customized and integrated IT solution service to most of their customers meaning that the service covers all stages of system development. More specifically, responsibility of Company N includes consulting service at the launch of new businesses, development of the software or system, and maintenance of the system. In addition, with many of the customers, Company N is under contract to develop and manage data centers as well. Accordingly, Company N has successfully maintained long term relationships with many of its customers.

As discussed in the previous paragraph, one of the advantages of Company N as a software firm is the long and firm relationship with their customers. In many cases, Company N manages development and maintenance of the entire system for their customers. In other words, many of the customers outsource their entire system to Company N. To appropriately manage such responsibility, although depending on the size of the outsourced system, Company N establishes independent department for each customers. By doing so, each

department successfully accumulates adequate knowledge regarding customers' business and operation so as to sufficiently support them as a true business partner. Deep understanding of customers' business has supported Company N greatly in achieving trust from their customers and to build long term relationship as discussed.

### ***3.2.2 Company philosophy***

To better understand fundamental business strategy of Company N, taking a moment to deliberate company philosophy is worthwhile. Out of several items mentioned as company philosophy, I would like to emphasize two points which significantly influence company's activities in software development projects.

First one is a phrase mentioned in the corporate mission which states that the company's goal is to "prosper together with customers". This simple philosophy is reflected significantly on how Company N operates its service to customers. As discussed previously, Company N is involved in all stages of software development starting from the planning and launch of a new business, development of the software itself, and maintenance of the delivered system. In other words, Company N has multiple opportunities to gain sales within different stages of software development. Out of these multiple opportunities, Company N depends on system maintenance to gain sales and the philosophy is the exact reason why because in the maintenance phase, it would be apparent if the new business was successful or not. To be more specific, Company N charges its customers based on the amount of data processed through the system, meaning that their profit depends on whether the delivered software or system was

useful or not. By adopting such business model, Company N and its customers establish a win-win relationship in the long run. Meanwhile, this has a substantial impact on how software development is handled because in most cases, major portion of software development costs are covered through profits made in the maintenance phase. Although budget planning and resource allocation is done for every software development phase, majority of the cost will be covered through turnovers in the maintenance phase. Unfortunately, this method frequently opposes ambiguity and laxness on cost management knowing that it will be reimbursed in through the maintenance phase. In software development, this causes a great difficulty in terms of freezing the requirement. Without definite limitations on the budget, customers continuously attempts to add new functions to the system and creates difficulty on project managers. Therein, appropriate control of system requirements is a challenge for Company N.

Another important philosophy pursued at Company N is for every employee to become a “true professional”. Due to the nature of the service which the company provides, training their employees to become professionals of what they do is crucial. Therefore, similar to how manufacturing company would invest millions of dollars on factories and new technologies, Company N spends substantial amount of its profit on training their employees. Newcomer training program is a good example showing how serious Company N is about training their employees. As discussed earlier, one of the unique characteristics of Japanese software industry is the scarce relationship between academic achievements of his/her school days and professional work. Without exception, many of the newcomers who enter Company N are absolutely new or an amateur to the discipline of software. To handle this reality, Company N executes extensive training course lasting 6 months for newcomers. Through this training

program, each employee achieves basic understanding of software development and acquires programming abilities in basic languages such as C, Java, and COBOL as well as fundamental skills as a business person. Newcomer training continues after 6 months by means of OJT where each newcomer is assigned to a well experienced instructor at their assigned departments. Primary objective of the OJT is for newcomers to acquire customer specific knowledge as well as to perform their programming skills in actual work environment. Although most of the programming is outsourced to partner companies which specializes in programming, newcomers spend their first year at Company N as programmers to comprehend software programming skills to prepare themselves to become professional project managers. In addition to the newcomer training, Company N continuously provides training courses which individual employees can attend based on assigned projects, degree of responsibilities, and personal interests. Through providing various well organized training programs, Company N trains every employee to become a true professional throughout their career.

It is interesting to discuss how recruitment and training style of the software industry of Japan affect some of the characteristics of its software development. As discussed previously, because many of the employees do not hold a computer science degree and are not programming specialists, it is unlikely that one of them will become so-called “super programmer” capable of creating innovative software product or leading the company with a novel development method. In the case of Company N, strategy is rather practical and it focuses on managing and organizing software development projects without taking significant role in the actual programming. Such strategy, to distinguish the labor of programming to outside partner companies, both domestic and international, is common in Japanese software

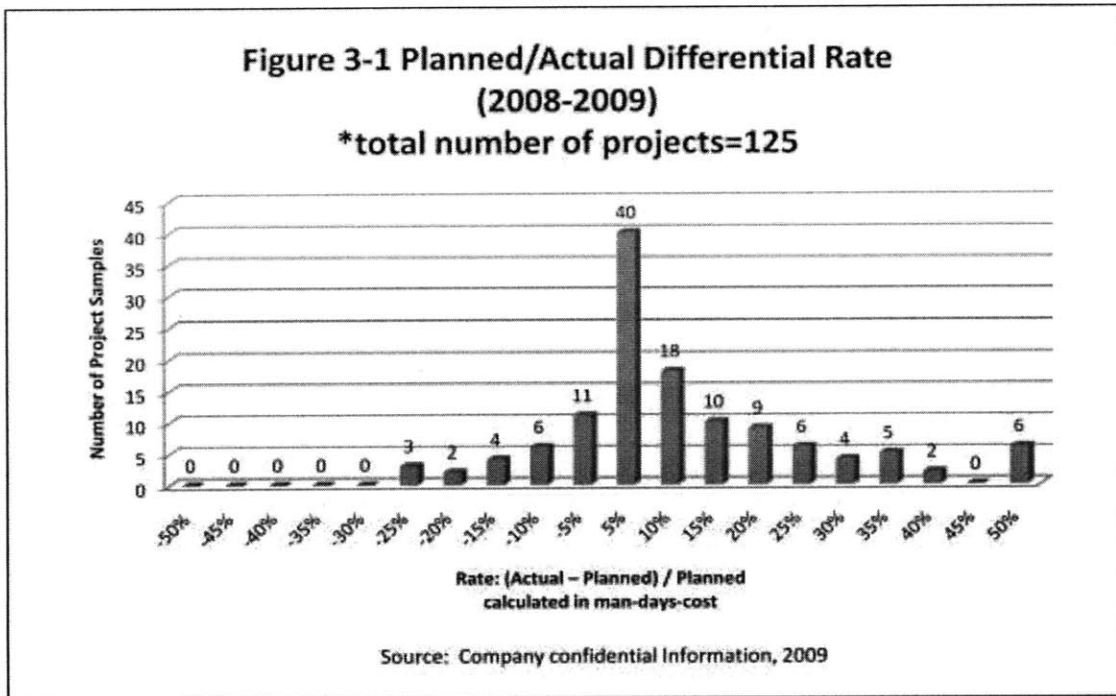
industry. This, in part, can be thought of as an attribute of the unique relationship between academia and business in Japan.

### ***3.2.3 Overview of software development by data***

To better understand the nature of software development practiced in Japan, particularly at Company N, couple of generic data indicating important factors of software development activities at Company N will be presented here. All of the data presented in this section is based on Company N's confidential information provided in the year 2009 as a summary report of various projects conducted in the previous years. Again, for anonymity, the title of this report will be covered.

One of the most frequently raised issue in software development is the difficulty of completing projects within allocated cost and schedule. Is this also true of Japanese software development projects? Below is a data describing the difference between planned and the actual man-days-cost, which essentially represents the man-power and time spent on various projects conducted between 2008 and 2009 at Company N. Total number of sample project included in this figure is 125. Note that the index is acquired through following calculation.

$$\triangleright \text{Planned/Actual Differential Rate} = (\text{Actual} - \text{Planned}) / \text{Planned}$$



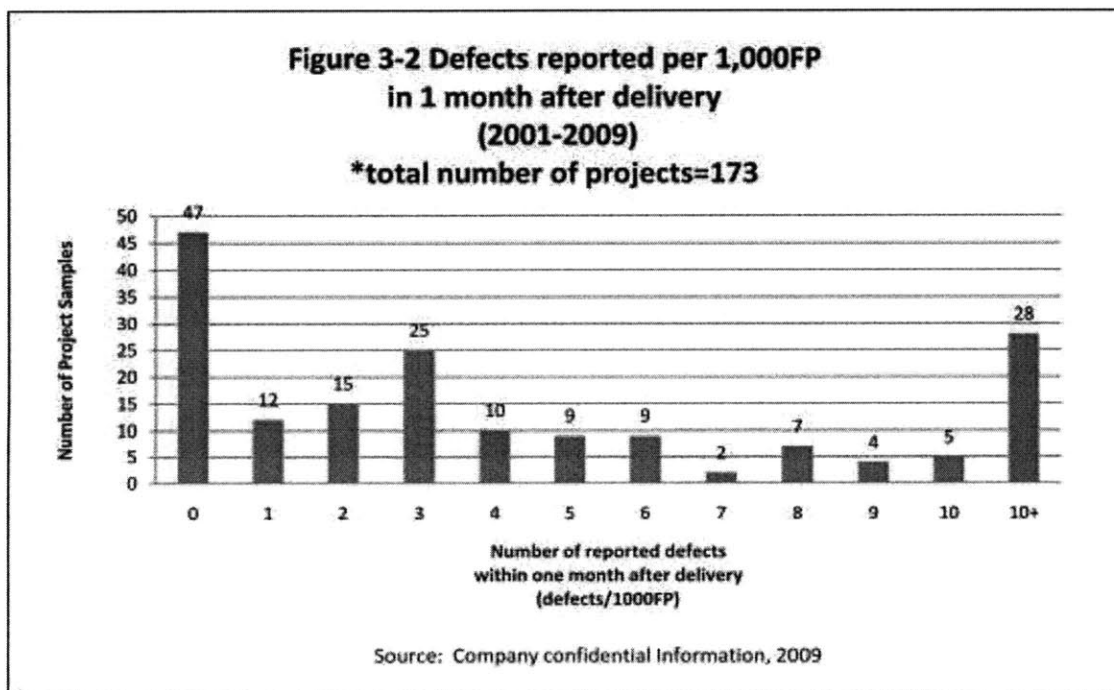
As shown in Figure 3-1, cost and schedule overrun is an apparent challenge at Company N with close to 80% of the project overrunning its initial plan. Moreover, with one out of four projects overrunning its plan by over 20%, it is critical as a software company to prove its cause and act to improve current development practices. In the same report, project managers reported some of the common reasons for such overrun as followed.

- Requirements were added by the customer after the onset of the project
- Ambiguous functional design lead to increased working hours
- Lack of appropriate outside interface definition lead to functional complexity and added to programming effort
- Insufficient review process in the functional design and detailed design caused programming rework

- Preparation and execution of the system test involved greater effort than initially planned

As represented in these comments, many of the problems arise from managerial difficulties rather than technical. This is exactly the reason why managing software development processes is crucial to a software company. By appropriately managing relations with customers, partners, and your own employees, software firms can resolve many of these problems.

Another interesting data to look at is related to the quality which, as previously discussed, is thought of as one of the strength of Japanese software industry. Figure below represents number of defects reported within one month after delivery at Company N.



Median of the numbers represented in Figure 3-2 is “2.5” defects per 1,000FP. This number, if converted to defects per 1,000 LOC (lines of code), is equivalent to “0.14”. According to the

“Software Kaihatsu Hakusho 2008<sup>3</sup>”, Japanese software industry in total is “0.18” for the same figure. Accordingly, Company N shows slightly higher quality level compared to other software firms in Japan. Looking beyond, in an article “Software Development Worldwide: The State of the Practice<sup>4</sup>”, Professor Cusumano and his colleagues reported worldwide comparison of similar figures. This article concludes that Japanese projects had one tenth of defects compared to projects in India and one twentieth of defects compared to projects in US. Although there may be differences in the nature of software development projects handled in each countries, which may be the cause to some extent, these numbers show apparent advantage of Japanese software development in terms of quality control.

Next data introduced represents the ratio of man-working-days devoted to each of the five stages of software development cycle, namely architectural design, functional design, detail design to unit test, system test, and user test. It also provides the ratio of outsourcing for each of the stages. Ratio of outsourced work is calculated by the following.

$$\text{Ratio of Outsource} = \text{Partner Man Working Days} / \text{Employee Man Working Days}$$

In other words, this number represents how many engineers are outsourced for every employee at Company N. Higher the number, greater dependencies on outside partner companies including both onshore and offshore.



**Figure 3-3 Effort spent per development phase and outsource ration**

	Architectural Design	Functional Design	Detail Design to Unit Test	System Test	User Test	Total
Ratio of Man Working Days spent on each Stage	7%	16%	37%	17%	23%	100%
Ratio of Time Spent on each Stage	16%	20%	25%	17%	22%	100%
Outsource Ratio	1.6	3.8	9.3	5.5	3.6	

Source: Company confidential Information, 2009

Couple of insights can be drawn from Figure 3-3.

- As commonly acknowledged within the software industry, greatest effort of work, close to 40%, is put on the actual development stage including detailed design, programming, and unit test.
- Despite the fact that many of the issues arise due to ambiguous requirement definition and changed requirements, man-working-days spent on upper stage of development are relatively low.
- In the upper stages of development, namely architectural design and functional design, ratio of man-working-days devoted are relatively low while time spent are equally distributed throughout the 5 stages. In other words, upper stage of development is completed by few engineers spending longer hours while the actual development involves greater number of engineers in order to shorten development time.

- Apparent dependency on outsource can be observed in programming and unit tests.

Meanwhile, low dependency for the upper stages and user test is naturally accepted due to high involvement of the customers.

As can be understood from these findings, appropriate management of different stages and efficient use of outsourcing is crucial for successful software development. In the next section, closer view of a particular project focusing on these matters will be presented.

### ***3.3 Description of a sample project***

To better describe the actual activities and relationships between various stakeholders involved in Japanese software development, analysis of a sample project will be presented. This section aims to describe fundamental activities crucial to software development projects through a sample project managed by Department R, one of the most profitable departments at Company N.

#### ***3.3.1 Overview***

Department R is responsible for managing consultation and system development services for a convenience store chain X. It consists of approximately 120 employees all devoted for this single client. Company N started its relationship with Client X, approximately 30 years ago and has strengthened its relationship over the years. This is a very good example of a success abiding by the company philosophy of “prospering together with customers”. Through 30 years of partnership with Client X, members of Department R are knowledgeable about

business of their customers and the system which support it. Client X, whose policy is not to own any infrastructure on its own, outsources management of the entire information system including development and maintenance. To handle such responsibility, Department R maintains various teams divided by system functionality from accounting, ordering system, distributions, and back-office support. In addition to Department R, Company N operates additional department responsible for managing data center for Client X. It is not too much to state that Department R operates as the information systems department for its client. Company N considers such long and firm relationship with various clients as the greatest advantage for a system integrator.

In the next four sections, we look deeper into specific activities at Department R with a focus on factors discussed in Chapter 2. Although it will only introduce activities of a single department, Department R being one of the most innovative organizations within the company, it will be a good representation of how software development is managed at Company N.

### ***3.3.2 Customer Relationship***

As noted earlier, Client X's basic principle is to not manage information systems on its own. Instead, it outsources development of information systems to system integrators such as Company N. To manage tight relationship with partner companies, Client X establishes information systems department whom accordingly is the primary stakeholder for Department R. Information systems department of Client X and Department R of Company N has established strong relationship over the years to enhance smooth operation and manages

various activities to support acquirement of business know-how. Such activities include sending employees to work at Client X's office for certain period of time, visitation of factories, and attending training programs hosted by one another. Through these activities, Department R continuously attempts to understand Client X's business and seeks new opportunities for system proposal.

In terms of software development, projects operate under clear division of responsibilities between the two companies. Although final decision making is, of course, made by Client X, significant part of the authority is given to Company N which shows great trust. Here is how development projects proceed. During the requirement definition phase, including architectural design, the two entities go through number of meetings to define basic requirements of the system. Meetings are held frequently, in most cases on daily basis, depending on progression of the project. At this stage, Client X takes initiative in describing their needs and the responsibility of Company N is to propose a functional and feasible plan to resolve client's needs. After the requirement definition stage, Company N begins the actual development of the software starting with the functional design and from this point up until the system testing phase, the only substantial involvement of Client X is the weekly progression check unless there are significant rework necessary regarding defined requirements. During the user test which is conducted after the system test, members from Client X are responsible for checking the usability and functionality of the system from user's perspective. At this point, in some projects, unfortunately, serious issues may surface such as misunderstanding of requirements, omission of requirements, and inappropriate functionality. Such problem regarding requirements definition leading to substantial rework is very common and most

troublesome in software development. In the case of Client X and Company N, such rework is in most cases resolved on Company N's expense meaning overwork for members of Department R and sometimes adding manpower temporary. However, in such cases, both parties are to be accused. First, improvement in accuracy of the requirement definition is necessary, second, client side needs to get more involved in the course of development in order to confirm functionality, and third, developer side needs to take initiative in confirming functionality with its client.

### ***3.3.3 Partner Relationship***

At Department R, most of the programming effort is outsourced to its partner companies in order to allow project members to focus on building client relationship and managing projects. Historically, Department R has been in long term contract with several partner companies which can act as the pillar of development projects. In addition to these main partners, short term contracts are concluded based on scale and schedule of project progression. All partners, long term and short term, are asked to be permanently stationed in the same building with which the members of Department R are settled in. This allows frequent reviews of the outsourced work. By contract, all programming efforts and unit tests are to be given final reviews by employees of Department R who are responsible for managing the project. Of course, partner companies conduct internal reviews before presenting to members of Department R. Such twofold reviewing processes play significant role in quality control by

ensuring fulfillment of the requirements presented by the client and checking for any flaws in the result of unit tests.

As noted earlier, primal responsibility of partner companies, in the case of Department R, are to program software based on presented functional design. However, as partner companies develop a long term relationship with Department R and acquire sufficient knowledge about Client X, they are asked to extend their roles. For example, some of the most experienced programmers of main partner companies are assigned leadership roles to manage other partner companies together with employees of Department R. As seen in this example, presence of partner companies, especially the main partners, are essential for Department R to successfully complete software development projects.

One of the recent issues in Japanese software development industry is the high cost of labor in Japan. According to the tight economical situations which many of the Japanese companies are facing, general investment for information system has fallen which has forced software companies including Company N to seek cheaper labor force outside of Japan leading to the trend of offshore development. Such trend, together with the high dependencies on domestic partner companies, have changed ways which Department R manages software development projects greatly.

#### ***3.3.4 Offshore development***

Department R, just as most of Japanese software companies were, started offshore development in the year 2000. Due to lack of English skills, cultural similarities, and geographical proximity, Company N has chosen Chinese software developer as offshore development destination instead of partnering with players of India. Such partnership strategy has been the trend of offshore development in Japan. According to an article presented by Sakura Kojima and Makoto Kojima, “Making IT Offshoring Work for the Japanese Industries<sup>5</sup>”, Japanese software companies tend to prioritize language proficiency when choosing offshoring destination and the fact that there were 63,000 Chinese students studying in Japan (in 2005) as opposed to 364 Indian students clearly explains why Japanese firms chose China as its partner.

In the same article, Kojima indicates the following as valuable lessons for successful offshore projects.

- Enhancing language proficiency and mutual cultural understanding is critical
- Investment in knowledge transfer education, especially at initial stage, is critical
- Regular execution of joint reviews are critical
- Long-term contract based on partnership may improve motivations of Chinese vendors

Interestingly enough, all of the lessons raised above are incorporated into Department R’s offshore development policies. To support Chinese engineers to learn Japanese and understand Japanese culture, Department R invites several members each year from China for a year-long training in Japan. During this training program, members from China not only brushes up their language skills but acquires knowledge about Client X and software development methods practiced at Department R which supports them significantly later on. Such exchange of

personnel happens both ways. In the initial stage of projects, Department R sends employees to China for the purpose of communicating software requirements and to transfer necessary knowledge required to execute development tasks. Once the programming tasks are on track, code reviews and test reviews are carried out on daily bases to complement for the risks of offshore development. In case if members of Department R are not at site, videoconferences are held using interpreter if necessary. In terms contracts, Company N is under long term contract with most of offshore development destinations in China. The objective of this long term relationship is to mimic the domestic long term relationship with Japanese developer which has been the motive power for Company N in the past.

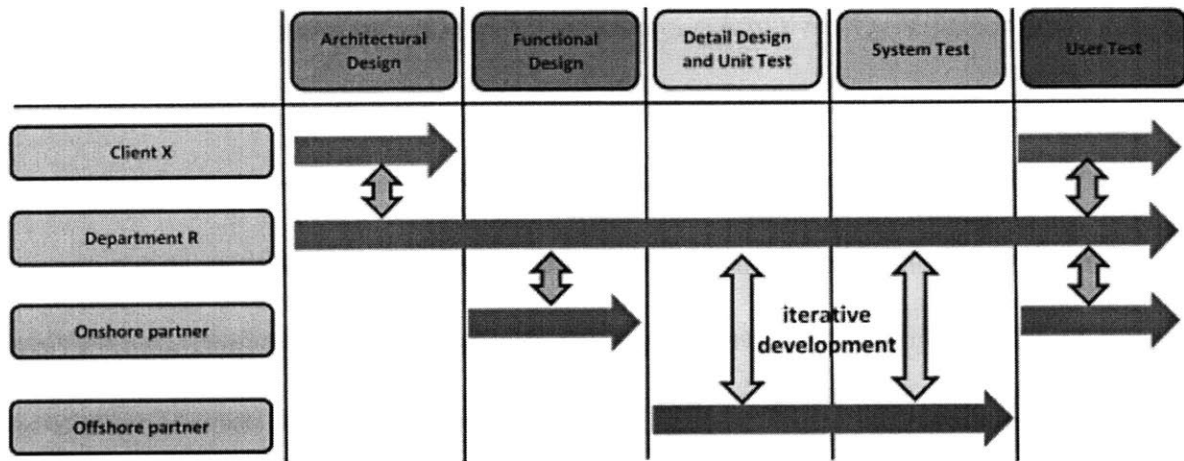
Despite all of the efforts, Department R has experienced some difficult issues related to offshore development. First, language barriers are inevitable which slows down development and possibly lead to substantial rework due to miscommunication of requirements. Second, quality degradations were verified due to low motivation and skills of offshore programmers. Third, lack of earnestness to understand customer's requirements were critical. To resolve these issues and benefit from offshore development, Department R seeks to establish some adjustments in current offshore development practices.

### ***3.3.5 Development Process***

With the explanation of current software development practice up to this point, it may be clear that Department R manages its software development under a waterfall model rather than an iterative development. As presented earlier, requirement definition is generally



conducted upfront between Client X and Department R. Then, based on defined system requirements, project managers undertake the actual development phase with its partner developers. During this phase, there aren't any significant interchanges between the client side and the developer side. Therefore, software is developed based on requirements defined upfront and if not specified in the requirements, judgment of members of Department R who is supposed to have sufficient understanding of Client X's business to make such decision. In a sense, development stage is operated in an iterative manner between Department R and partner developers, particularly with the offshore partners. However, in general, development processes are designed based on the premise that no rework will occur between individual development stages. To illustrate this premise, explanation of documents that Department R is responsible for submitting to Client X is lucid. At the end of each stage, Department R delivers various documents to Client X. Such documents include, "requirements definition", "functional design definition", "system test report", "user test planning", and "user test report". Each of these documents needs to be approved by Client X before moving on to the subsequent stage in development cycle which clearly shows character of a waterfall model development. Figure 3-4 is a representation of development model practiced at Department R showing active players involved in the development.



**Figure 3-4 Task responsibilities and interchanges between parties**

While Department R is involved throughout the development process as sort of a control tower, other players have specific stages they are responsible for. One of the problems with such method is that extra effort is required to explain system requirements and expected tasks every time new players get involved. For example, Department R must explain system requirements to both onshore and offshore partners along with the explanation of project progression up to the previous stage of development.

As it is for most of the software firms, which operates a waterfall model development, greatest issue for Department R is the change of requirements in the later stages of development. It is not rare for Department R to face significant rework after the testing phases. However, it is almost as if members have reconciled to it because in most cases, Department R complies with the request for modifying requirements. This is probably because of the vocation to protect the long term relationship with Client X and also because of the “customer first” state of the mind ingrained in Japanese culture. Anyway, it is essential for Department R to establish more efficient development style in order to compete in this difficult economy.

### **3.4 Challenges**

In this section, to close this chapter, issues inhabited in software development at a typical Japanese software firm will be stated as a summary.

As repeatedly mentioned throughout this chapter, biggest challenge for Department R is how to avoid changes in requirements and handle them in the event it occurs. This probably is a common challenge for many of the Japanese firms, especially those operating under a waterfall model development. In the present state of affairs, while acknowledging some of the weaknesses of this model, many of the Japanese software firms cope with it in order to minimize cumbersome interchanges between various players within the project and to maintain trustworthy relationship with their clients. As a result, however, in many cases result in significant cost and schedule overruns. To successfully provide software development and system integrating services in the recent cost driven and global environment, Japanese software firms must alter current practices and establish more efficient way of managing software development projects.

In addition, as high cost structure of Japanese labor becomes outstanding and dependency on inexpensive offshore labor force become increasingly apparent, stronger relationship with Chinese and other new emergent software developer will become significant. Accordingly, Japanese software firms are urged to improve current practices to overcome communication and cultural barriers. Alongside, clear division of responsibility between

onshore and offshore partners to bring out best overall performance will be an important task for software firms like Company N.

Next two chapters will focus on software development practices of United States and India. What are the differences? What are the similarities? What can be applied to improve current software development practices?

# Chapter 4: Reference Study – US Firm

---

As mentioned in the introduction of this paper, one of the objectives of this thesis is to analyze Japanese software development in contrast to those of leading players in the industry, namely United States and India. This chapter focuses on the former, software practices in US software firms, discussing some of the important characteristics commonly observed along with two of the important software development concepts, agile software development and capability maturity model, which originated from software practices in the US.

First section of this chapter focuses on two of the most famous, profitable, and influential firms in US software industry, namely Microsoft and Google. One of the facts that are commonly acknowledged within the global software industry is the supremacy of US firms in the production of packaged software and internet service. In that sense, focusing on these two companies would successfully capture one of the important characteristics of US software industry. Microsoft started its business in 1975 and expanded steadily synchronous to the spread of computer usage during the last 35 years. Literally, it has been the leading software company in the world, showing overwhelming presence among the industry and has become a role model for software firms around the world. Meanwhile, Google which was established in 1998 has acquired top share in search engine business and its method of managing software development is another focus of interest among software industry. Although these two companies have contrastive history as software firms, there are some interesting commonalities in its strategies and principles that are driving their successful business. Through

looking at these commonalities, this section aims to organize some of the important characteristics common to, or vision for many, software firms in the US.

Second and third section of this chapter focuses on two of the important concept in software development, namely agile development and capability maturity model, in some details. Both of these concepts, which originated from software development methods in US, are commonly practiced in many software firms around the world today and are worthwhile to analyze its effectiveness and compatibility to Japanese software development. For these two sections, some of the obstacles or challenges in implementing these concepts to Japanese software development will be discussed alongside.

#### ***4.1 Basic strategies of software development at US firms***

Before going any deeper into the discussion of some interesting strategies practiced at US software firms, it is important to understand and acknowledge the fundamental differences between Japanese software firms and those of US. As described in the previous chapter, many of the Japanese software firms, including Company N, focuses on providing customized system or software solutions to their clients. Meanwhile, US software firms has its strength in packaged software such as Microsoft Windows and internet service represented by Google. Consequently, it is natural to recognize differences in its development methods between the two groups. However, it would still be valuable to analyze and understand the advantages of practices in US firms and attempt to apply them at a Japanese software firms.

As mentioned in the introduction of this chapter, we will be looking at Microsoft and Google as representatives of US software firm in this section. In order to cover the lack of practical knowledge on either of these two companies' development practices, facts and thoughts discussed in this section are based on reference study from the following two literatures. For information regarding Microsoft, a book by Professor Cusumano, "Microsoft Secrets"<sup>1</sup>, and for Google, a book by Girard, "The Google Way"<sup>2</sup> is referred.

#### **4.1.1 Hiring Policy**

Hiring policy of a software company, hence the group of people working there and the sort of mindsets they possess, has a significant impact on how that company operates its business. Without exception, software development practices are greatly affected by company's hiring policy. Hiring policy of Microsoft and Google is very similar, both declaring that they are focused on hiring smart people, the best available candidates. Remarks made by Bill Gates, then CEO of Microsoft, in 1993 during an interview represents such policy very well.

*"The key for us, number one, has always been hiring very smart people. There is no way of getting around, that in terms of I.Q., you've got to be very elitist in picking the people who deserve to write software. Ninety-five percent of the people shouldn't write complex software."<sup>3</sup>*

Both Microsoft and Google strive to recruit employees who fulfill their expectations through various hiring policies.

One of the beliefs in hiring new employees at Google, in contrary to that of Japanese software firms, is that people who continue their studies at graduate level are not only more intelligent and better trained but are more impassioned and motivated in their field which makes them a better candidate to work at firms like Google. In the terms Girard uses to describe hiring policy at Google, “recruiting people with graduate degrees is a way to hire those who are highly motivated and value the quality of their work above their immediate personal interests.” In addition, according to Girard, Google believes that graduate students, who generally possess the experience of conducting a graduate-level research of their chosen topic, are capable of demonstrating their abilities to successfully lead an innovation at a high tech company like itself. Consequently, Google focuses on hiring candidates who possess graduate-level degree in the field which accords with the corporate strategy at the time.

The policy of hiring smart people sounds even more explicit in the case of Microsoft as words from Bill Gates quoted in “Microsoft Secrets” implies.

*“We benefit from having the smartest people actually get involved not just in the design and ideas but actually the code itself. They know the code extremely well, and large areas of the code. It’s good to have one person who can think through any kind of design change, particularly when you’re late in the process and you’re trying to be very careful. They can review anything that goes on. They’ve got a very strong model in their head for what...side effects that change might have.... We have the benefit that people don’t like to have bad developers around.”<sup>4</sup>*



As can be seen from this remark, Microsoft believes strongly that hiring smart people with appropriate background can establish groundwork for efficient software development. In this regard, Cusumano points out how some of the software firms including many Japanese software firms may hold a contradictory view in hiring policy. These companies choose to hire people from various backgrounds and with minimal experience in software discipline to train them from scratch so that they can influence them in the way they prefer.

These two characteristics in hiring policy is influential, beyond all expectation, in terms of establishing other important characteristics of US software firms as it will be discussed later in this section. After all, significant portion of software development is managing group of engineers to work efficiently towards a common goal and the way a company manages engineers depends on the mindsets and motivations of their employees.

#### ***4.1.2 Working in small teams***

Both Microsoft and Google represent the effectiveness of working in small teams when developing software products and in fact, according to the literatures, they do manage software projects in small teams of 6 to 10 engineers. Conducting software development projects in small teams has advantage in various ways in that it creates environment where both management and programmers are forced to work productively and efficiently. First of all, because each team is small and resources are limited, management is forced to assign projects with precise goals and short easily monitored deadlines. Precise goals allow project teams to work without time loss and increases productivity. Also, because the deadlines are easily

monitored, problems are easily detected and resolved promptly. Second, by working in small teams, overhead caused by communication difficulties and cumbersome coordination among team members is prevented. In addition, small team prevents freeloading because each member's performance is easily observed and peer pressure act as a monitor leading to efficient advancement of the project. Through making the most of such advantages of working in small teams, Microsoft and Google successfully manages software projects with high productivity and efficiency continuously providing innovative products to the market.

Although operating software projects in small teams may sound like an answer for every company, it is important to acknowledge that the quality of individual employees is essential for success. Therein, policy of hiring smart, well motivated, and disciplined employee is a strategy which needs to be considered alongside and that may be the exact reason why it was an effective practice for both Microsoft and Google.

#### ***4.1.3 How innovation is inspired***

While hiring smart motivated employees and putting them into small teams to collaborate and compete against each other already builds innovation prone environment for the company, Google has another unique practice which inspires innovation, called the "20 percent policy". Here is an explanation by Girard of this policy.

*"Google's stated policy splits the work hours of its engineers and developers into two parts: Eighty percent of their time is dedicated to assigned projects, the*

*official source of their paycheck, with the remaining 20 percent dedicated to personal research of their own choosing<sup>5</sup>".*

Google acquires several advantages from this policy. First, they are able to attract brilliant engineers who prefer to obtain autonomy to some extent and the liberty of continuing their work based on personal interests. Secondly, the policy enforces employees to work efficiently and productively on their assigned work in order to allocate additional time on their personal work. Thirdly, which is more of a direct advantage, research and work of individual employee may lead to the emergence of new products for the company. In fact, some of the important product already introduced to the market is actually an outcome from the "20 percent policy". For the three reasons discussed, this policy has been extremely effective for Google.

In addition to an internal motivating factor, so to say, there exists an external factor which inspires employees to strive for innovation at any company in any industry. Easy to imagine, earning money is the greatest motivation for many employees. This is true especially in the software industry where many of the firms are capturing great business opportunities to amass great riches in significantly short period of time. To appropriately relate business opportunity, profitability, and employee motivation, many companies are granting stock options while keeping regular salary relatively low as represented in the case of Microsoft. By placing large portion of reward on stock options and bonuses that is provided proportional to his or her accomplishments, positive attitude for seeking innovations and new business opportunities is ingrained within the company. Such way of providing incentives for employees can be extremely effective for a software company whose core competence is in providing

package software products capable of dominating the market. As we all know, Microsoft has literally dominated the world with software series such as Windows and Office achieving enormous amount of fortune for the party. Thus, knowing how successful their precursors were, Microsoft employees lavish greatest efforts everyday to seek new innovations. Here, linking back to the fact that Japanese software firms specializes in customized software rather than packaged software targeted for the mass market, penetrating inspiration for new innovation among employees is rather difficult to achieve. In other words, effort of individual employees does not directly link to company's profitability because each employee's work is deeply related to respective clients. Subsequently, company's atmosphere is far from encouraging innovations and even in the case where it occurs, it hardly ever links to company's new business.

## ***4.2 Agile software development***

Now that we have looked into some of the fundamental characteristics of software development of US software firms, this section focuses on one of the relatively recent trend in software development, agile software development which can part be thought of as a derivative of those characteristics discussed in the previous section. Although, Microsoft and Google are good representatives of firms who operate agile software development, in some cases expressed as iterative development, this section will discuss agile software development in general.

As repeatedly mentioned in this paper as well as commonly acknowledged by software engineers, the main cause of cost and schedule overrun in software development projects is

the frequent change in software requirements. Although some thoughtless engineers might blame this solely on clients' indecisiveness or lack of business strategy, in today's rapid evolving business environment, engineers must realize how fast it is evolving and learn to cope with these changes. Agile software development practices were developed to do just that. Mike Holcombe, in his book, "Running an Agile Software Development Project"<sup>6</sup>, describes features of agile software development as the following.

*"Any agile software development process has to be able to adapt to rapid changes in scope and requirements, but it has also to satisfy the needs for the delivery of high-quality systems in a manner that is highly cost-effective, unburdened by massive bureaucracy, and that does not demand heroics from the developers involved."*

As implied in the statement above, key to a successful agile software development is to have frequent communication with the client in order to understand up-to-date requirements and also to establish flexible organizational and cultural structure to appropriately handle frequent changes to build on-demand software.

#### **4.2.1 Extreme Programming**

For the purpose of achieving those ultimate goals discussed in the previous section, software firms have thought out and practice numerous agile methodologies including dynamic systems development method (DSDM), feature-driven design, agile modeling approach, SCRUM,

and extreme programming (XP). While all of these methodologies all are based on agile philosophy and all have its strengths and weaknesses, by far, XP has been the trend in the software industry. For that sake, we will look at XP approach, as one of the representative of agile software development, briefly but in some details. Note that the section is referring to Holcombe's publication<sup>7</sup>.

#### **4.2.1.1 XP Values**

Based on the notion that software development is all about human activity, there are five fundamental values which exist as the basis of XP method. The five fundamental values include issues around communication, feedback, simplicity, courage and respect. Although, upon hearing these five values, it is not easy to understand how they relate to software development, an engineering discipline, they surely are important factors for successful software projects. Here, brief discussion of how they can be important is provided.

##### ***-Communication***

Communication is by far the most important factor for any software development projects to be successful. In fact, many of the failure in software projects can be attributed to a breakdown in communication between developers and clients, among the clients, and among the developers. This is especially true for an extreme programming where frequent changes in requirement are anticipated. First, clients must make frequent communication among themselves to determine principal objective of the software which corresponds to the current

business environment. Second, clients must communicate to the developers any necessary changes to the initial requirement definition as soon as possible. Third, developers need to communicate among themselves in order to incorporate new requests from their clients. Communication must flow accurately and smoothly in all three dimensions in order to appropriately manage extreme programming based projects.

### ***-Feedback***

Feedback is another important concept of extreme programming which in a way must be thought of as part of the communication process. It is significantly important for the purpose of maintaining vigorous and beneficial communication for the project. For example, in order to effectively keep the client involved in the project, it is important for the developers to inform how they are handling change of requirements, how the testing is proceeding and progression of the project so as to allow clients to give feedbacks. Only with such feedbacks, developers are able to make additional modifications to improve the software to fulfill client's true needs.

### ***-Simplicity***

In multiple means, simplicity is valued in extreme programming. Firstly, design simplicity is valued in order to avoid surplus functionality which leads to unnecessary complexity of the software. Both clients and developers are responsible for questioning whether the planned design appropriately captures business needs. Secondly, excessive effort during the design phase should be avoided in extreme programming because changes in requirements are anticipated. Accordingly, concise designing which allows clients to provide proper feedbacks

while keeping design effort on the developer side to a minimum is favored. For example, heavy documentation in the design phase is avoided.

### ***-Courage***

Courage in the context of extreme programming means to hold willingness to adapt to the client's changing needs as the project develops. Undoubtedly, it takes courage to invalidate what you have already developed and start from the beginning. In other words, the philosophy of extreme programming asks engineers to relish new challenges.

### ***-Respect***

Lastly, respect is the underlying core of the extreme programming philosophies. As implied from the previous four values discussed, extreme programming is only achieved through frequent and effective interactions among various stakeholders. To be successful, all of the stakeholders involved must share common understanding of the process and collaborate as a project team. Therein, treating individual with respect is essential to cultivate environment where members communicate freely to express feelings, understand one another, and take necessary responsibilities.

#### ***4.2.1.2 XP Practices***

Based on the five philosophies discussed, extreme programming method encourages 12 practices as discussed below. Since there are numerous documents published regarding these



practices with excellent and concrete examples, this paper will only provide a list with short explanation for each.

### ***-Test-First Programming***

In extreme programming, developers are encouraged to build a set of tests before writing any code. This allows programmers to test software based on its functionality which matters the most for the client rather than focusing on the structure or how it was coded.

### ***-Pair Programming***

All programming efforts are made by pairs of engineers. While one person uses the keyboard to write-up the code, other person constantly checks for logical errors and seeks for possible improvements. This process of continuous review contributes to less frequent mistakes and also provides valuable learning environment for the programmers.

### ***-On-site Customer***

XP method encourages clients to be on-site which enable frequent and face-to-face communication between clients and developers. By minimizing communication overheads, development processes can be accelerated significantly.

### ***-The Planning Game***

XP software development is based on business stories provided by the customer or small pieces of meaningful functionalities. Throughout the development cycle from cost planning, coding phase, and testing, developers work in iterative cycle based on these business

stories which allows them to stay focused and in control to accumulate efforts while allowing appropriate involvement from the clients.

### ***-System Metaphor***

Metaphor is used simply to facilitate transmission of the business stories or the software requirements. If used appropriately, metaphor can diffuse and develop common understanding of what their ultimate goal is among clients and developers.

### ***-Small Frequent Releases***

Another important practice of the extreme programming is to release early and release often which provides the opportunity for clients or users to provide feedbacks. Based on feedbacks provided, developers will make necessary improvements to make another quick release. Such iterative process, probably the greatest feature of agile development methods, allows incremental improvements of the software.

### ***-Simple Solution***

As discussed earlier, one of the philosophies of extreme programming is to keep things simple. Accordingly, developers must always seek to keep software simple functionally, technologically, and structurally. Ultimate question is, “does the customer really need that function?”

### ***-Continuous Integration***

For the purpose of promoting incremental and frequent releases while appropriately controlling the entire development activity, codes under development are integrated into the system at least a few times every day. For every integration process, unit tests for individual module as well as functional test for the integrated software are conducted. This in a way is to secure a working group of modules on daily basis.

### ***-Coding Standards***

Building software based on shared coding standards throughout the development process promotes consistency in terms of quality and also allows developers to easily review codes and sometimes to modify them. In addition, it contributes to robust interfaces between various modular. These advantages are significant in XP based projects where frequent changes in parts of the software are anticipated.

### ***-Collective Code Ownership***

As a result of applying coding standards, all members of the development team are able to understand any parts of the software written by another programmer. Therefore, XP development projects operate under the concept of collective ownership of all codes. Essentially, any programmer can modify any parts of the software at any time to improve it as a whole.

### ***-Refactoring***

XP project teams are encouraged to continually restrict code without changing the functionality of software for simplicity. Through simplifying codes, program sources become

more understandable, and thus easier to maintain or modify for future expansion. Simplicity of source codes is considered as an essential factor to accomplish speedy and high quality development.

### ***-Sustainable Development***

XP, through the five concepts and twelve practices, aims to eliminate various stresses from members of the development team. Encouraging lively communication and punctual feedbacks, especially, are significant factors in creating good working environment for the team to effectively collaborate and take control of the project. Consequently, as result from past examples prove, XP based projects rarely get out of control to evolve into a “death march” projects.

#### ***4.2.1.3 How effective is XP?***

As commonly acknowledged within the software industry, extreme programming method is not applicable to all types of project. For example, large projects involving multiple developers and numerous engineers may not be a perfect fit for XP method. Due to significant importance of close and frequent communication among all stakeholders involved, having a large party simply makes it difficult. In addition, client-developer relationship and organizational structure of the project team is an issue for successful XP projects. First, unless good relationship is maintained between the client and developer, active communication and feedback will be impracticable. Second, relationship within development team is another

important factor. For example, if the project relies on offshore labor force, some of the practices may need to be devised. Therefore, managers at software firms must realize that XP method is not a perfect remedy for every software projects.

In the case of common Japanese software firms, as discussed in the previous chapter, it may require reformation of mindsets to some extent in order to switch gears from a traditional waterfall model development to an agile development. Both clients and developers including offshore partners must alter their current practices to effectively take in concepts of XP. Nevertheless, some of the practices of XP may not be effective in some cases due to peculiar constitution of Japanese software firms. For example, necessity of frequent client-developer communication might be thought of as a burden for many of the clients in Japan because they are so much accustomed to confiding entire development to an outsource. However, with all that said, although it may require some modification, basic philosophies or concepts advocated in XP can be effective in many situations. Tendency to prioritize long term relationship with clients, as seen in many Japanese firms, may even support promotion of active communication. After all, managers of software firms all need to acknowledge that today's business environment is evolving so quickly that traditional waterfall model cannot cope with its speed. Therein, it is vital for software companies to adopt, or at least consider agile development concepts.

#### **4.2.2 Scrum**

One other agile development method commonly practiced in US software firms, which is very similar to XP, is the SCRUM. While the focus of XP is in engineering point of view, SCRUM puts greater emphasis on management perspectives of software development. Some important concepts of SCRUM are presented below.

### ***-Sprint***

In a SCRUM, software projects are divided into small functions or features of software each with assigned deadline and costs. Such divided features are then developed individually in cycles or iterations called “sprint” which are kept short usually 30 days, in some cases as short as 2 weeks. And at the end of each sprint, intended features are reviewed and confirmed for fulfillment of the requirements.

### ***-SCRUM meetings***

For the purpose of monitoring progress of each tasks or sprint closely, daily meetings are held. The point of scrum meeting is to keep it short, usually under 15 minutes, and focused on reporting progression of each tasks and occurrence of problem if any. In case there is a problem, deeper discussion is conducted outside of SCRUM meetings to avoid any overheads.

### ***-Backlog***

Product backlog and sprint backlog plays important role in a SCRUM to manage requirements of the product and tasks for particular sprint respectively. Backlog is usually maintained using wiki pages accessible and editable by all project members. Such high accessibility facilitates participation of each member greatly mitigating project management

efforts significantly. In addition, use of backlog creates a type of autonomy to some extent in terms of task assignment by allowing project members to acknowledge what needs to be completed next.

As seen from the three important methods discussed, the important concept of SCRUM is to keep iteration cycles short, avoid any management overheads, and allow engineers to exercise their abilities in order to achieve high efficiency and productivity. Such concept seems to be in line perfectly with the strategies of US firms to hire smart people and keeping project teams small. In a sense, SCRUM is a method which can bring out best performances from group of pure engineers to resolve some of the difficulties in software projects. A quote from an article by Striebeck describes the situation very well.

*With the help of an experienced agile leader it was possible to carefully introduce agile practices into Google – an environment that does not have an affinity to processes in general.....All this could be done without destroying the great bottom-up culture that Google prides itself of. The practices only affect how the projects are structured. Design and implementation remains fully an engineering responsibility<sup>8</sup>.*

Concepts of SCRUM, along with practices included in XP, are very much worthwhile for Japanese software firms to consider. It may be a good starting point to introduce agile concepts in the management perspective as seen in the SCRUM to make a transition from current waterfall model development.

### **4.3 Capability Maturity Model (Integrated)**

*CMMI is a process improvement approach that provides organizations with the essential elements of effective processes that ultimately improve their performance. CMMI can be used to guide process improvement across a project, a division, or an entire organization. It helps integrate traditionally separate organizational functions, set process improvement goals and priorities, provide guidance for quality processes, and provide a point of reference for appraising current processes.*

This is an excerpt from Software Engineering Institute's website<sup>9</sup> providing an overview of the Capability Maturity Model Integration (CMMI). CMMI, a successor of the well known CMM developed in the late 1980's, is a framework for organization seeking to improve business processes and hence its overall performance. Note that in this paper, CMM and CMMI will be mentioned interchangeably due to relatively recent appearance of CMMI. While there are many frameworks aiming for the same objective, CMMI is outstanding for two reasons. First, CMMI is superior in terms of coverage in various aspects of activities involved in managing a company. It covers activities in project management, engineering, support, and process management which are all relevant in the context of today's complex working environment. Second, CMMI successfully provides guidelines for companies seeking for process improvements, in terms of what aspects in current activities they should focus next. In other words, it indicates what steps to take in order to move on to the next level. CMMI recognizes companies into 5 levels, known as the maturity levels,



according to the types of activities ingrained to the organization. Brief essentials of the 5 levels are presented below followed by figure 4-1, an illustrated representation of the maturity levels.

**-Level 1 (Initial)**

Processes are undefined, chaotic in many cases. Organization at this level relies on heroic employees for project success.

**-Level 2 (Managed)**

Projects are effectually managed and documentations such as project plans, requirements definitions, and review reports are filed properly.

**-Level 3 (Defined)**

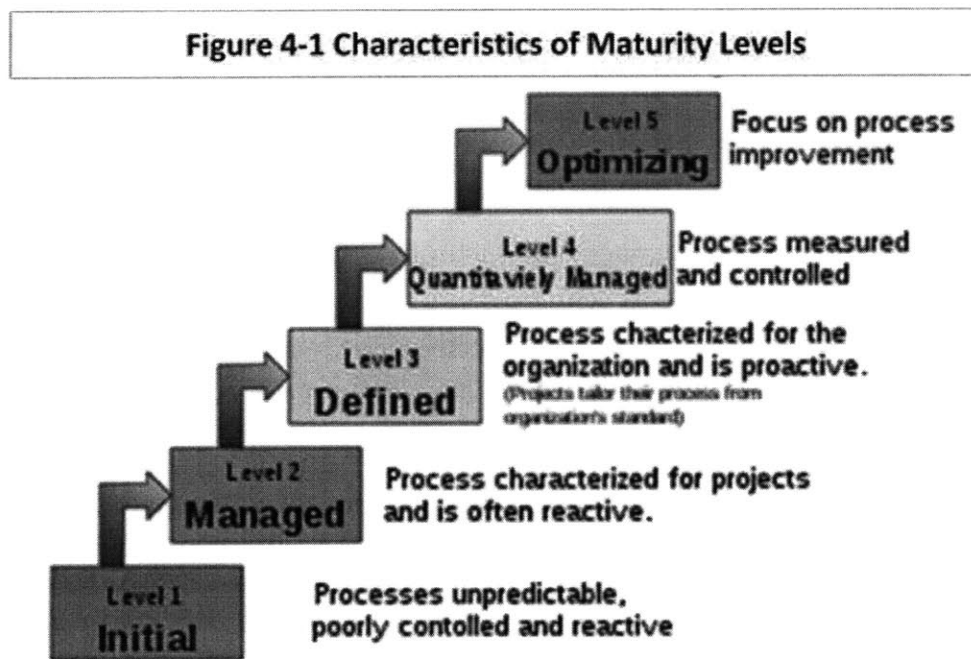
Standard process is defined and applied throughout various projects within the organization. Individual projects are tailored and managed based on organizational standard process.

**-Level 4 (Quantitatively Managed)**

Statistical and quantitative measures are collected and used for criteria in managing projects. Greatest distinction from level 3 is that at level 4, managers successfully use these figures to make quantitative predictions on process performances.

**-Level 5 (Optimizing)**

Level 5 organizations are capable of continuous improvement of its business processes. Process improvement objectives and goals are properly defined as well as altered promptly depending on changes in the business environment. In addition, result of any process improvements are measured quantitatively to reflect on further improvements.



Source: [http://en.wikipedia.org/wiki/Capability\\_Maturity\\_Model\\_Integration](http://en.wikipedia.org/wiki/Capability_Maturity_Model_Integration)

For each of the maturity levels discussed, CMMI defines desired activities or processes in four areas of process improvement as explained earlier. Explanation of these activities incorporated in each level is excluded in this paper. Readers interested in learning in depth may refer to CMMI for Development<sup>10</sup>.

Reports from organizations which has applied CMMI into their activities and followed the 5 maturity level guidelines prove the effectiveness of CMMI in various indices,

namely cost, schedule, productivity, quality, customer satisfaction, and return on investment. Along with such direct benefits, some indirect benefits are also observed such as improvement in quality awareness and providing training opportunities for prospective project managers through clear and logical conduct of a project. Overall, firms that have adopted CMMI have reported positive feedbacks regarding its effect on project operations.

CMMI, for some software firms, is thought of as sort of demonstration to show its ability to manage software projects and fulfill clients' needs. From the standpoint of clients who assign projects to outside software firms, CMMI can be an excellent benchmark to assess their abilities. In the case of US software industry, especially in the defense industries, some clients oblige their potential outsource-destinations to acquire certain level of CMMI certification. Although this example may be on the extreme, it is true that many software firms around the world is applying CMMI to their organization, acquiring appraisals, and most importantly realizing its effectiveness. Figure 4-2 is an excerpt from a presentation in 2009 by Software Engineering Institute of Carnegie Mellon University reporting count of CMMI levels certified around the world. Although numbers only include officially reported cases, there are approximately 3,000 companies involved globally and of those companies, over 70% are non-USA organizations. As can be seen from the numbers, China and India, two of the most active and relatively new players in the software industry are affirmative in acquiring CMMI certification besides United States. Apparently, major players in the global software industry are showing interest in adopting CMMI. It is interesting to note that, India, which will be discussed in the next chapter, has the greatest number of CMMI Level 5 certified organizations. This, in a way, explains their significant presence in the global software market.

Figure 4-2 Number of Appraisals Reported to SEI by Country

Country	Number of Appraisals	Maturity Level 1 Reported	Maturity Level 2 Reported	Maturity Level 3 Reported	Maturity Level 4 Reported	Maturity Level 5 Reported	Country	Number of Appraisals	Maturity Level 1 Reported	Maturity Level 2 Reported	Maturity Level 3 Reported	Maturity Level 4 Reported	Maturity Level 5 Reported
Argentina	69		48	13	2	4	Malaysia	59		20	34		5
Australia	94	1	7	7	2	4	Mauritius	10 or fewer					
Austria	10 or fewer						Mexico	69		29	30	3	5
Bahrain	10 or fewer						Morocco	10 or fewer					
Bangladesh	10 or fewer						Nepal	10 or fewer					
Bolivia	10 or fewer						Netherlands	10 or fewer					
Belgium	10 or fewer						New Zealand	10 or fewer					
Brazil	117	1	57	45	1	9	Norway	10 or fewer					
Bulgaria	10 or fewer						Pakistan	25	1	19	4		1
Canada	55	1	14	23	5	3	Panama	10 or fewer					
Chile	94		20	11		2	Paraguay	10 or fewer					
China	946	1	122	728	30	45	Peru	10 or fewer					
Colombia	24		7	12	2	2	Philippines	22		2	11		8
Costa Rica	10 or fewer						Poland	10 or fewer					
Czech Republic	10 or fewer						Portugal	10 or fewer					
Denmark	10 or fewer						Romania	10 or fewer					
Dominican Republic	10 or fewer						Russia	10 or fewer					
Egypt	58	1	20	12	2	2	Saudi Arabia	10 or fewer					
Finland	10 or fewer						Singapore	20		3	11	1	4
France	153	4	59	50	1	2	Slovenia	10 or fewer					
Germany	70	5	34	13	1	1	South Africa	10 or fewer					
Greece	10 or fewer						Spain	155	1	59	48	3	4
Hong Kong	18		2	11		5	Sri Lanka	10 or fewer					
Hungary	10 or fewer						Sweden	10 or fewer					
India	460		14	206	24	172	Switzerland	10 or fewer					
Indonesia	10 or fewer						Taiwan	124	1	75	34		2
Ireland	10 or fewer						Thailand	34		12	20		1
Israel	18		3	10		3	Turkey	15			13		2
Italy	37		18	16			Ukraine	10 or fewer					
Japan	290	18	82	131	13	16	United Arab Emirates	10 or fewer					
Korea, Republic Of	147	1	50	55	14	7	United Kingdom	100	3	45	32	1	3
Latvia	10 or fewer						United States	1405	27	469	538	21	131
Lithuania	10 or fewer						Uruguay	10 or fewer					
Luxembourg	10 or fewer						Vietnam	14			11	1	2

Source: <http://www.sei.cmu.edu/cmml/casestudies/profiles/pdfs/upload/2009SepCMMI.pdf>

In the case of Japan, although there is number of firms adopting CMMI, its emphasis is relatively low and hence is not a priority for most firms. As can be seen from Figure 4-2, in Japan, only 5% of the reported appraisals acquired maturity level 5. While this could be an indication of the immaturity of Japanese software firms, here it is not the case. Rather, it shows seriousness of an organization as a whole in challenging to acquire CMMI certification. Overwhelming ration of Level 5 organization in India was a fruit of efforts of the entire Indian software industry. It is also true that there are numerous oppositions to the effectiveness of CMMI accusing its bureaucratic and impractical methods. Therein, general understanding within Japanese software industry is that while CMMI can be an indicator of systematic approach of an organization to improve its competencies, CMMI

alone cannot act as a barometer to see the capacity of any organization. In addition, as for Japanese software firms, several reasons for the slow adoption of CMMI can be raised. First, the fact that many of the Japanese software firms are focused on customized software/system has an impact. Due to a rather passive relationship with its clients, each software projects tend to adapt clients' way of business respectively. Consequently, process improvement activities are less likely to develop into an organizational movement as encouraged in CMMI. Second, lack of connection between academia and commercial sector is limiting CMMI recognition. While CMMI is a common topic in the field of computer science, many of the Japanese software engineers do not have an idea of what it actually is. In addition, lack of researchers and experts in the field leads to scarce number of authorized appraisers capable of assessing for CMMI appraisals in Japan. With that said, one of the aspects of Japanese software industry favorable in terms of adopting CMMI is that Japanese engineers culturally excel in following guidelines and rules as discussed in the previous chapter. In that sense, if an organization can properly adopt CMMI and establish its process improvement methods, it can be an effective tool in Japan as well.

# Chapter 5: Reference Study – India Firm

---

Today, India, along with China, is recognized as one of the most important player in the software industry. With its abundant talent with software specialties, Indian software firms are in a favorable position to dominate the global IT market. In a study, India's software industry is indicated as having acquired 1.5% share in the global IT market in the year 2000 and projected to have grown to approximately 5% today<sup>1</sup>. This is an extraordinary figure not only in terms of India's significant presence in the global market but also in terms of the speed of its growth. The important question is how they were able to acquire such success in this short period of time which is the question this chapter aims to answer.

Here is an interesting phrase commonly acknowledged by people involved in the business of software which describes the nature of India's software industry. "India is an IT service center of the world" as opposed "China is a factory of the world". In other words, significant part of India's share in software industry deals with value-added upstream portion of the software development while China has not been able to eradicate its position as destination for low-value downstream outsourcing. IT related service provided by Indian software firms extends from upstream consulting service to the downstream application outsourcing service. Also, industries served by Indian software firms are well diversified including manufacturing, banking, insurance, telecom, retail, and transportation. Therein, Indian software firms literally cover the entire IT service depending on the needs of their clients.

Referring back to the objective of this paper, it is worthwhile to explore software firms of India both in order to understand the strategies which enabled them to be so successful in the global market and also because of the similarities between Indian and Japanese software firms. According to Kojima<sup>3</sup>, over 30% of the Indian export related to software service deals with customized application development and its maintenance. Therein, focus of software firms in the two countries coincide with one another which it to provide customized software solutions for their clients. In that sense, it is worthwhile for Japanese software firms to refer to the strategies pursued by the Indian firms.

In the first section of this chapter, characteristics of Indian software firms which as it stand are its advantages over other competitors in the global market will be discussed. Then, in the second section, we will be looking at the top three software firms of India, namely Tata Consultancy Services (TCS), Infosys Technologies Limited (Infosys), and Wipro Technologies Limited (Wipro), to understand some of the basic strategies of Indian Software firms. Interestingly, as mentioned by Professor Cusumano in his article “Envisioning the Future of India’s Software Services Business<sup>3</sup>”, with an apprehension for Indian software industry, the three companies are very similar in terms of its strength and what they are offering to its clients. Therefore, although the section focuses on top three players in India, it may be thought of as a representation of the entire Indian software industry.

### ***5.1 General advantages of Indian software firms***

Annual report of the second largest software firm in India, Infosys, indicates three key factors contributing to the growth of Indian software industry<sup>4</sup>. Here, we look briefly into these key factors.

### ***-High Quality Delivery***

It is a well know fact among software industry that India has the greatest number of firms certified at SEI-CMM Level 5. In the year 2008, 158 out of 362 firms which were certified Level 5 appraisal were from India. Although, as discussed in the previous chapter, CMM solely cannot evaluate abilities of a company entirely, it can be thought of as one of the signs for a high quality delivery. By the way, all three top players which will be discussed in the following section are CMM Level 5 organizations. In addition to the high ratio of CMM Level 5 certified firms, reference study conducted for the purpose of this thesis has proven impressive efforts of Indian firms to improve quality of the deliverables so as to maximize customer satisfaction. Some of these efforts will be discussed in the following section in some details.

### ***-Significant Cost Benefit***

Many of the companies which choose to confide software development tasks and IT related services to Indian firms claim greatest advantage of doing so as the significant cost benefit. According to the Infosys's annual report, such companies experience 60-70% cost reduction through outsourcing IT related services to India. Although this figure may include some dramatization and may be weakening as India's economy grow and lives of the people of India enrich, cost benefit will remain as the greatest advantage of working with Indian software firms and will continue to be for the coming several years. Moreover, as discusses in the



opening of this chapter, fascination of India's software firms is that they are able to provide value-added upstream service at such a low cost.

### ***-Abundant Skilled Resources***

The most significant factor why India's software firms were able to establish presence in the global market and continue to grow at a remarkable speed is the large and highly skilled English speaking labor pool. As it will be discussed in the following section, India's software industry excels in securing superior talent both through academia-industry coordination and on-the-job training. To borrow the exact words from Infosys's annual report, "the large and growing pools of skilled professionals has been the key driver of the rapid growth in the Indian IT-ITES sector and that India accounts for over 28% of the total suitable talent pool available to work in the IT-BPO sector across all the potential global sourcing low-cost locations<sup>4</sup>". That is a significant figure for one country to be responsible for and is a representation of how abundant India's software resource is.

These three advantages, common in most software firms of India, together with the trend of placing greater emphasis on IT outsourcing has boosted up India's presence in the global market. In today's highly evolved, globalized, and competitive situations, it was a natural stream for companies to focus on its core competence and attempt to outsource supportive projects such as software development to low-cost developers. Together with the high fluency in English language, India has become the primal destination for Western countries to

outsource IT related businesses. In that sense, India has successfully captured the global trend and adopted needs of their client to establish strong presence in the software industry.

## ***5.2 Business Strategies of Indian software firms***

In this section, we will be looking into practices pursued in top three software firms of India in some details. These top three players, namely TCS, Infosys, and Wipro, has lead the software industry of India over the past several decades and now has established significant presence in the global market. While the three firms differ in course of establishment and varies in its strength to some extent, basic approach in pursuing its software business is surprisingly similar. Furthermore, it is a surprising fact that somewhat similar strategies are practiced at many of the software firms in India. Accordingly, this section will attempt to provide explanation to some of these basic approaches pursued at Indian software firms through representations of the top three players.

### ***5.2.1 Tata Consultancy Service***

Tata Consultancy Service (TCS), which was founded in 1968, has the longer history than any other software firms in India and is recognized as the pioneer and a leader within India's software industry in terms of overall sales, exports, and number of software engineers employed. It is not too much to say that TCS, literally, has lead the evolution of software industry in India. TCS, like many of the software firms in India, covers broad range of clients

including finance, manufacturing, transportation, telecom, and retail. It also provides diverse types of IT related services covering the entire value chain. Although it specialized in low-value downstream services during the early years of its history, TCS is now recognized as one of the leaders in high-value upstream consulting company in the global market. Another interesting characteristic of TCS is its emphasis on exporting service. In the year 2002, ratio of export to the entire sales was over 90% while that of Indian software industry was approximately 80%<sup>5</sup>. The basis of its success lies in its rigorous hiring policy, excellent training program, and outstanding quality management conducted by such superior personnel.

#### ***-Human Resource Management***

With no doubt, abundant supply of human resources with deep IT knowledge has been the key factor of India's success. While the ratio of students enrolled in high degree education such as universities and colleges to the total population is extremely low at approximately 5.8% compared to most of the developed countries, simple head count of the total number of students enrolled is ranked second place after United States. India, by far, has been number one in Asia in terms of high level education until China has accomplished marvelous growth and took place of India's position. Nevertheless, India still remains on top among Asian countries in the engineering discipline especially in software and IT related fields. Indian software firms have successfully utilized such abundant source of talent to explore opportunities especially in the international market to establish its current preferable position.

TCS, as discussed earlier, number one player among India's software industry, is in a favorable position among domestic competitors to attract excellent talents due to its high

reputation. According to Kojima<sup>6</sup>, TCS hires approximately 30% of graduates each year from Indian Institute of Technology (IIT), which is well recognized as the most prestigious engineering school in India which well describes TCS's attitude towards hiring the best people. But hiring the best people is not the only secret to success. TCS spends approximately 4% of its sales each year on training its employees. For example, new hires experience over 100 days of rigorous training program before they are issued an assignment. During this training session, employees go through series of lectures and programs to acquire practical knowledge and skills which are expected to become each employees' foundation of their professional career. Themes brought up in these lecture series include concepts in software engineering, systems engineering and concurrent engineering, and basic business skills such as communication, presentation, negotiation, team working, and cross cultural collaboration. As can be understood from such curriculum, TCS attempts to provide thorough training opportunity for its new hires to become a successful business person in the field of IT. In particular, emphasis on concurrent engineering, team working skills, and understanding cultural differences indicates TCS's serious attitude towards achieving success in the global market. Apparently, such efforts have paid off for TCS.

Yet, another interesting fact about the source of human resource at Indian software firms, including TCS, is the number of returning professionals and students from US. While Indian labor force has played significant part in the growth of US IT industry for several decades, it wasn't until recently that these labor force has started to return to India for new opportunities. Due to both the downturn of US economy and upturn of India's software industry, preference of people who were employed by US software firms and students studying in US Universities have shifted to exploring opportunities in India from acquiring a position in

US firms. Such trend in the flow of human resource have provided Indian software industry with software development know-how practiced in US software firms and personal connections with executives which largely contributed to the expansion of full-outsourcing contracts from Western countries especially from United States.

### ***-Quality Management***

Although the cost benefit of outsourcing IT related services to India has played a significant factor for firms in Europe and US to partner with Indian software firms, this alone could not have achieved the great success of Indian software firms in the global market. Another significant factor lies in high quality software solutions and services provided by the Indian software firms. As noted earlier, India has the greatest number of firms that are certified CMM Level 5 which means that these firms are able to use quantified metrics to evaluate its current development processes and make systematic improvements within the organization. More specifically, indices such as the productivity and defect rates are quantified to assess current capabilities of an organization. Such assessment, along with evaluation from clients, is used to derive measures to improve current affairs or processes on a daily basis. Consequently, CMM Level 5 organizations are able to make continual and systematic process improvements.

TCS, just as many of the software firms in India has done, acquired CMM Level 5 in many of its branches. Through its efforts to acquire CMM Level 5 appraisal and as result of process improvement activities encouraged by CMM, TCS has experienced improvements in various dimensions. Such improvements include more accurate estimation of cost, decline in project management costs, decrease in schedule overruns, and substantial reduction of reworks.

Overall, customer satisfaction and productivity has improved significantly allowing TCS to achieve more contracts with new clients leading to its flourish. As can be seen in this example, quality management has been a great factor for India's success in the software industry.

### **5.2.2 Infosys Technologies**

Infosys Technologies, the second largest software firm in India has aimed to become global company from its establishment in 1981. As a proof of that, it was the first software firm in India to be listed in NASDAQ market. Although its business was limited to low-value downstream portion of the value chain at start, typical to firms of IT emerging countries, Infosys has now established high-value upstream position and calls themselves "IT Consulting Service Provider" with a sense of pride. Today, under the highly globalized and IT oriented environment, Infosys has successfully established relations with the world's first-class companies in various industries and through the course of interaction with such customers have acquired knowledge and technologies of the leading-edge business models. Furthermore, Infosys has successfully nurtured such knowledge as its own strength for further expansion.

#### **-Global Delivery Model**

Global delivery model (GDM) is a strategy pursued by most of the Indian software firms which enables clients from around the globe to benefit from high-value-add low cost service. Basic notion of the GDM is to structure and utilize globally distributed teams to provide customers with low-cost on-demand IT solution service. GDM allows software firms to appeal

to its clients, the low cost realized through the use of cheap labor force based in India while maintaining tight relationship at the site of their clients' which allows them to understand the needs of their customers and provide appropriate solutions. Therein, it pretty much seems to be a reasonable strategy for Indian software firms that are highly focused on providing IT solution outsourcing service to pursue GDM. Infosys is one of the pioneers of this concept and has succeeded greatly in the global market through its execution. According to Infosys<sup>7</sup>, benefits provided through the execution of GDM are as followed.

- Access to large pool of highly skilled technology professionals
- 24-hour execution capabilities across multiple time zones
- The ability to accelerate delivery times of large projects by simultaneously processing project components
- Built-in redundancy to ensure uninterrupted services
- A knowledge management system that enables re-use of solutions

A typical GDM at Infosys is conducted with a combination of onsite and offsite teams collaborating as a project team. In the earlier phases of development where project scope and requirements are defined, members of the project teams are sent to the client's site usually from an office located in the same graphical region called global development centers. Once the system requirements are defined, center of the action is shifted to offsite locations where labor costs are relatively cheaper, usually branches in India. Although most of the work in the development stage is done offsite, members at the global development center are standing by to visit clients whenever necessary to enforce smooth execution of the project. Not only does

establishing global development center located in close range allow quick response to clients' needs but also mitigates cultural differences through establishing similar mindsets. In addition, valuable asset in terms of knowledge of clients' business is accumulated to establish long-term relationship.

What is remarkable about GDM pursued by Indian software firms is that while operating multiple global development centers in various locations, it is able to maintain control as a project team and perform effectively while benefiting from its advantages. For example, one of the things mentioned as an advantage of GDM, the simultaneous processing of project components in various locations, is not an easy thing to accomplish which requires developers to overcome difficulties of establishing consensus of the system, making distant communication, and overcoming cultural differences. Yet, Indian software firms have successfully implemented GDM and are providing high-quality service around the world. There are two factors which can be explained as keys to India's success in GDM. First, most importantly, is the cross organizational efforts of establishing standard procedures, frameworks, and methodologies as represented by its efforts in CMM Level 5 acquisition. As explained earlier, CMM Level 5 companies are able to make continuous improvements to suit changing business environments and optimize its activities. In the case of Infosys, such effort is conducted at every development center which is then integrated as an activity throughout the organization in many cases establishing into company standards. Through such endeavor, members located at various global development centers are able to function seamlessly as a project team to provide flexible and optimal solution to its clients. Secondly, project management method and tools utilized cannot be neglected which we will discuss in the next section.



### **5.2.3 Wipro Technologies**

Wipro Technologies is unique in terms of its course of business evolution. It first started as a manufacturing company producing commodity products such as soap and cooking oil but later turned its wheels around to get involved in computer manufacturing business. It was in 1994 for that Wipro finally decided to focus its resource to software related businesses. Despite its late entry into the software industry, Wipro has shown miraculous growth to achieve third largest sales in India. Today, along with TCS and Infosys, Wipro is recognized as world's top class software solution provider. Among the three firms, Wipro was the first to acquire CMM Level 5 appraisal which may be resulting from its emphasis on process improvement possibly originating from its manufacturing background. In this section, we will be looking at two of the project management techniques practiced at Wipro which is a significant factor how software firms of India are able to manage GDM and succeed in the global market in general.

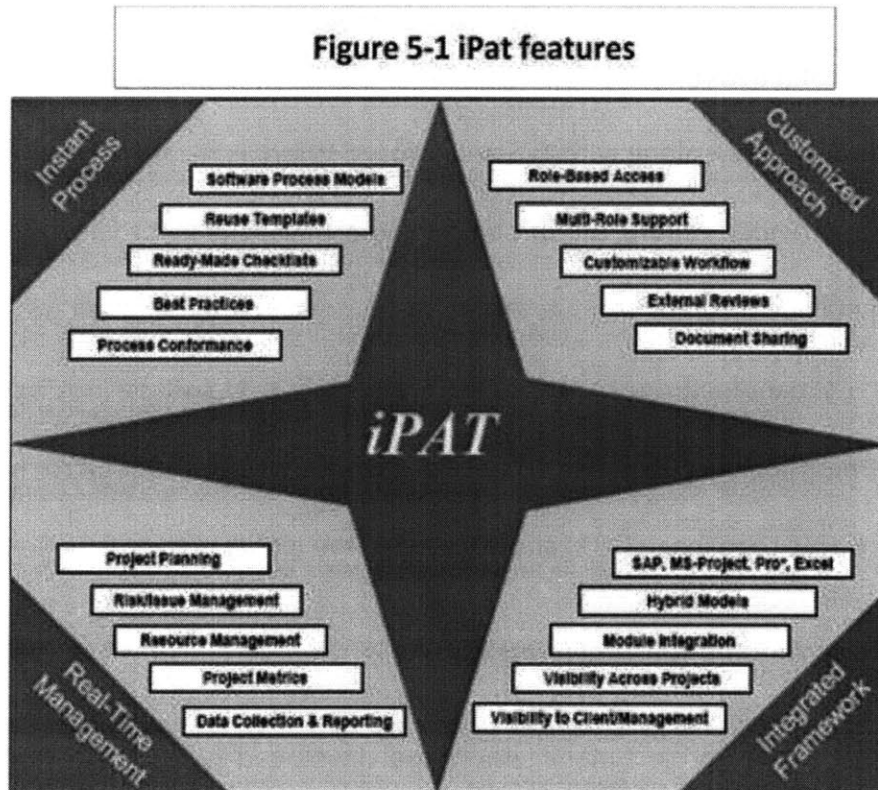
#### ***-Integrated Process Automation Tool (iPat)***

iPat is a web-based project management tool developed in-house by Wipro for the purpose of supporting project execution. All members of Wipro can easily access iPat through any web browsers. This powerful management tool plays a significant role to achieve success in the globally distributed projects which we have discussed in the previous section. In a report<sup>8</sup> written by group of researchers at Software Engineering Institute, main objective of iPat are explained as improving project management in the following factors.

- Senior management visibility of project status
- Effort in documentation and collection of data

- Effort spent in consolidating metrics data
- Resource utilization
- Consistency and accuracy of data

Here is a graphical representation of features and benefits of iPat.



Source: <http://www.sei.cmu.edu/reports/04tr006.pdf>

As depicted in figure 5-1, iPat is used for various objectives and situations. For example, it is used as a storehouse of project management related know-how such as templates, checklists, and standardized frameworks which project members can utilize whenever necessary. It is also connected to human resource database of Wipro allowing project managers to search available employees with sufficient skills when forming a team at the start of a project. In the initial stage

of a project, software requirements are loaded on to iPat so as to build common understanding among project members. Once the project starts, iPat also functions as a tool to manage project progression. All project members wherever they are located are required to report progress of assigned tasks or any problem if any through iPat on daily basis. Based on such information, iPat is capable of generating project progression report for the senior management to review. Accordingly, through the use of iPat, managers are able to detect project delays quickly and acknowledge what is causing it to make necessary measures. This is an extremely powerful function in the context of executing GDM projects to mitigate difficulties of geographical separation. In addition to managing project progression, iPat also functions as a tool for collecting quantified data such as defect rates and productivity which are used for objective decision making. Not to mention that these data can be used for CMM related activities as well. Words used by Hamm<sup>9</sup>, explains the significant outcome of iPat on Wipro's project management activities. "iPat has brought cultural change to the engineering groups. Suddenly, project management was driven by data, not the gut feelings of program and project managers".

Application of iPat is not limited to internal use. Project progression report on a monthly basis is provided to clients through Internet as well. In the standpoint of clients outsourcing critical software development projects to Wipro, such information is invaluable in terms of assuring project progression. This is especially true in executing GDM projects where conducting frequent meetings with the client is impractical. Above all, its appreciation is well reflected in the improved customer satisfaction.

### ***-Customer Satisfaction Survey***

At Wipro, customer satisfaction survey is an important part of project management process. For every project, upon completion, detailed feedback from manager on the client side is solicited. Also, in addition, second survey is conducted after a certain period of time to find out if the project had paid off for the client. Results of both surveys are taken seriously to improve current practice. In case of poor survey scores, the entire organization including executive members collaborate to find out what caused it and to come up with a solution to improve it. Furthermore, in order to ingrain the culture of pursuing customer satisfaction, result of surveys are considered heavily when evaluating project managers' annual performance appraisals and pay raises. Although many software firms conduct customer surveys including those in Japan, degree to which such surveys are valued is much less. In that sense, it can be said that obsession for customer satisfaction is another important characteristic of Indian software firms.

As discussed throughout this chapter, strength of software firms of India lies in the abundant low-cost labor force and project management capabilities to organize them efficiently at high quality standards. Although the history of these Indian software firms are relatively short compared to rest of the IT advanced nations, its significant efforts have paid off. With such strength, they continue to stand as attractive outsourcing destination for firms in Western countries especially United States. Accordingly, many of the Indian firms are establishing IT partnership with world class players in various industries. In the standpoint of Japanese

software firms, whose primal occupation is in providing customized IT solution, achievements of Indian software firms are of great reference. Although there are several apparent differences in the situation and background of the two countries, many of the concepts pursued at Indian firms can be considered for application to improve current practices.

# Chapter 6: Summary and Recommendation

---

This final chapter of this paper will summarize the findings discussed in previous chapters and based on these findings provide some recommendations for Japanese software firms.

## ***6.1 Summary of conditions surrounding Japanese Software Firms***

In chapter 3, we have discussed some of the unique characteristics of Japanese software firms which may have affected its current software development practices. Also, in chapters 2, we have discussed some of the transitions occurring in the software industry along with the picture of business environment. Then we looked at some of the competitors and their practices in chapters 4 and 5 which have contributed to clarify strengths and weaknesses of Japanese software industry. To summarize the findings of this research, key characteristics and current issues of Japanese software industry are presented here.

### **-Emphasis on customize software**

As discussed in chapter 3, many of the software firms in Japan focus on providing customize software for a particular client rather than producing package software for general consumers. Due to such characteristic, Japanese software firms are facing challenges accompanied by customize software such as the difficulty of capturing clients' business needs. Despite numerous efforts to mitigate such difficulty, misunderstanding

of software requirements and significant amount of rework is imposing serious problem to Japanese software firms.

#### **-Long-term relationship with customers**

Establishing long term relationship with clients has been a priority in Japanese software firms which seem to be in line with the fact that its focus is on providing customize software requiring deep understanding of client's business and needs. While long-term relationship brings various advantages for software firms, it also involves some disadvantages. In order to maintain long-term relationship with clients, Japanese software firms rarely have the luxury of rejecting requests for change in requirements. Also, long-term relationship brings a type of mannerism for project members which deprive opportunity to seek innovation.

#### **-Pursuit of waterfall model development**

Due to the two points discussed previously, Japanese software firms tend to practice waterfall model development. In most cases, software firm is expected to take full responsibility throughout the development cycle. In a sense, mitigating clients' burden by minimizing their involvement is believed to be an important value in Japanese software industry. Accordingly, however, in terms of software development, maintaining productivity and meeting deadlines has always been an issue at Japanese software firms.

#### **-High cost structure**

High cost structure of Japanese software industry is undoubtable. According to a research paper<sup>1</sup>, labor cost for mid-career software engineer in Japan is 12 times greater than China and 20 times greater than India. With such high cost disadvantage, it is critical for Japanese software firms to partner with offshore developers for cheaper labor force.

Apart from characteristics of software firms, we discussed some of the important trends and transitions occurring in Japanese software industry.

#### **-Importance of offshore development**

High cost structure of Japan necessitates use of offshore labor force for software firms. Today, as discussed in Chapter 3, many of the Japanese software firms are relying on Chinese partners for cheap labor force and such trend is expected to continue for the next several years. Offshore development, while providing significant cost benefit, imposes significant coordination efforts on software firms. Therefore, it is absolutely critical for Japanese software firms to establish more productive and efficient ways of managing offshore development projects.

#### **-Entrance of global players to Japanese market**

Traditionally, Japanese software industry was rather a closed market due to Japan's peculiar culture, language, and style of business. However, as our society became globalized and companies became more standardized, entering Japanese market has



become an option for global software players. For example, couple of Indian software firms, which we have discussed in Chapter 5, has already opened up branch offices in Japan. Moreover, it is only natural for Chinese software firms which have acquired business know-how of Japanese industry from past collaboration to enter Japanese market to become a direct competitor. Accordingly, Japanese software firms must acknowledge that the competition is intensifying.

#### **-Saturation of Japanese software industry**

In addition to competition intensifying, it is also true that Japanese software market is saturating. According to Katakame<sup>2</sup>, average annual growth rate of the Japanese information service industry was 14.2% during the years between 1997 and 2002 which has dropped dramatically to 4.8% between 2002 and 2007. Although the slowdown of software industry is not limited to Japan, it is apparent that Japanese software industry is starting to saturate. Therefore, it is imperative for Japanese software firms to seek business opportunities outside of Japan.

#### **-Evolution of new technology**

Introduction of new technologies, as represented by cloud computing, will certainly bring another shift to the software industry. The speed at which new businesses evolve and innovate will become increasingly faster which would require software firms to be more productive, flexible, and cost effective. In order to thrive in such environment, it is important for software firms to anticipate such change early on and take necessary measures.

## **6.2 Recommendations**

Keeping in mind the characteristics and current business environment of Japanese software firms as summarized in the previous section and also referring to some of the practices pursued by software firms of US and India, this paper provides four key recommendations for strategies Japanese software firm may pursue in order to survive the intense global competition.

### **1. Hiring a good mix of specialists and generalists**

One of the characteristics of Japanese software industry discussed in Chapter 3 was the lack of coordination between the academia and industry. More specifically, Japanese software firms are hiring many employees without computer science or any scientific background and spending substantial amount of time and cost to train them. On the other hand, literature reviews have shown that software firms in US and India are striving to hire excellent talent with special backgrounds to form small but extremely productive teams to manage software development projects efficiently. Such hiring strategy seems very reasonable for firms in these two countries in terms of the type of service they provide. Referring back to characteristic of Japanese software industry and its future, hiring a good balance of computer specialists and generalists capable of managing large projects is recommended. While technical innovation such as the introduction of cloud computing require software firms to secure talents with technical knowledge, higher dependency on offshore development and the trend for global expansion imposes greater managerial

challenges. To deal with such challenge, Japanese software firms must consider hiring generalists with excellent communication and managerial skills capable of managing projects on a global scale.

## **2. Adoption of agile software development**

While there is a good rational, as discussed previously, for Japanese software firms to prefer waterfall model development, it is imperative that they adopt more agile development practice. Disadvantages of waterfall model development are imposing serious problems for both software firms and clients which are expected to aggravate as firms are exposed in much greater competition both domestic and international. The recommendation, more specifically, is for Japanese software firms to promote combination of frequent onsite communication and offshore development which allows them to maintain excellent relationship with their clients while conducting agile development. Allocating personnel onsite of the client will allow software firms to better understand their needs and to make frequent communication which will facilitate agile development. It is also important to include offshore site in the agile process by making frequent communication. Consequently, greater consideration and efforts must be put on project management.

## **3. Standardization and continuous process improvement**

Although this paper does not necessary advocate for the application of CMM, significant role it played in the success of Indian software firms is unquestionable. What was observed from conducted research on India is the importance of establishing standards to be applied throughout the organization and to foster culture of continuous improvement which in the case of Indian software firms were achieved through CMM. One of the problems in

Japanese software firm is that distinctive departments are establishing its own way of developing software which best suit respective clients. Consequently, software development practices at Japanese firm lack consistency as an organization. As a result, coordination between departments is laborious and relocation of personnel imposes significant overhead of getting them accustomed to new development practices which makes it impossible to manage multi-location project teams as we saw in India's successful Global Delivery Model. Therefore, Japanese software firms must facilitate coordination among departments so as to establish standardized development practice. Subsequently, well standardized development process will allow Japanese software firms to make continuous and systematic improvements organization wide.

#### **4. Global expansion**

Due to the slowdown of domestic IT market and intensifying competition, Japanese software firms must seek global expansion. With its ability to produce high quality products and provide minute service, as seen from the success of manufacturing industry, Japanese software firms are capable of acquiring a favorable position in the global software market where they will compete against global players such as those of India. Therein, the greatest barrier is the high-cost structure which is a clear contrary to India's low labor cost. To mitigate such disadvantage, Japanese firms must utilize offshore development. In a sense, Japanese software firms must extend India's GDM one step further to incorporate offshore developers into the picture and make up for India's low labor cost. More specifically, recommendations is for Japanese software firms to establish development centers in offshore development destinations such as China and Vietnam which can provide cheap

labor force while geographical and cultural proximity is obtainable. The important thing to keep in mind is to implement a system that does not spoil values and practices fundamental to Japanese software firm. Characteristics of Japanese firms that were discussed in this paper are the very essence of such values and practices which derive its core competencies. Therefore, Japanese software firms must invest in hiring and training local people to convey those mindsets and values to operate these development centers. Meanwhile, in terms of building client relations, the recommendation is to send personnel from head office in Japan to establish long-term relationship just as observed in the domestic market. It is important to note that the value proposition made by Japanese firms in this context is not a low-cost delivery but rather a high value-add minute service at a comparable price. After all, due to high labor cost, Japanese firms cannot compete in the common ground as India. Although this extended GDM model, so to speak, may involve tremendous amount of management efforts, it is essential for Japanese firms to realize globalization in order to stay competitive and establish its reputation in the global market.

As readers may have noticed, all four recommendations are interrelated and cannot be accomplished individually. If Japanese software firms can execute and realize all four recommendations that were mentioned here, they will certainly be able to manage complex globally distributed projects to compete against global players. Overall, such extraordinary management capability will become the very value Japanese software will appeal to the world.

# Bibliography (by Chapter)

## Chapter 1

1. [http://www.standishgroup.com/newsroom/chaos\\_2009.php](http://www.standishgroup.com/newsroom/chaos_2009.php)

## Chapter 2

1. Makoto Shirota, "Kulaudo no Shougeki" (The impact of cloud computing), Toyo Keizai Shinpousha, 2009

## Chapter 3

1. Michael A Cusumano, *"The Business of Software: What Every Manager, Programmer, and Entrepreneur Must Know to Thrive and Survive in Good Times and Bad"*, Free Press, New York, 2004, p11-12
2. Michael A Cusumano, *"The Puzzle of Japanese Software"*, Communications of the ACM, Vol.48 No.7, July 2005
3. Software Engineering Center, *"Software Kaihatsu Hakusho 2008"* (Software Development Report 2008), Nikkei BP, 2008
4. Michael A Cusumano, Alan MacCormack, Chris F Kemerer, and William Crandall, *"Software Development Worldwide: The State of the Practice"*, IEEE Software, November/December 2003, p2-8
5. Sakura Kojima and Makoto Kojima, *"Making IT Offshoring Work for the Japanese Industries"*, First International Conference, SEAFOOD 2007, *"Software Engineering Approaches for Offshore and Outsourced Development"*, p67-82

## Chapter 4

1. Michael A Cusumano, Richard W. Selby, *"Microsoft Secrets: How the World's Most Powerful Software Company Creates Technology, Shapes Markets, and Manages People"*, Free Press/Simon & Schuster, New York, 1995
2. Bernard Girard, *"The Google Way: How One Company Is Revolutionizing Management As We Know It"*, No Starch Press, 2009
3. Bernard Girard, *"The Google Way: How One Company Is Revolutionizing Management As We Know It"*, No Starch Press, 2009, p54

4. Michael A Cusumano, Richard W. Selby, *"Microsoft Secrets: How the World's Most Powerful Software Company Creates Technology, Shapes Markets, and Manages People"*, Free Press/Simon & Schuster, New York, 1995, p69
5. Bernard Girard, *"The Google Way: How One Company Is Revolutionizing Management As We Know It"*, No Starch Press, 2009, p64
6. Mike Holcombe, *"Running an Agile Software Development Project"*, John Wiley & Sons, 2008, p2
7. Mike Holcombe, *"Running an Agile Software Development Project"*, John Wiley & Sons, 2008, p19-39
8. Mark Striebeck, "Ssh! We are adding a process...", Proceedings of AGILE 2006 Conference, IEEE Computer Society, 2006
9. Software Engineering Institute, CMMI overview, <http://www.sei.cmu.edu/cmmi/index.cfm>
10. Software Engineering Institute, CMMI-DEV, V1.2, <http://www.sei.cmu.edu/cmmi/tools/dev/index.cfm>

## Chapter 5

1. Kojima Makoto, "Indo no Software Sangyo" (Indian Software Industry), Toyo Keizai Shinpousha, 2004, p13-17
2. Kojima Makoto, "Indo no Software Sangyo" (Indian Software Industry), Toyo Keizai Shinpousha, 2004, p51-54
3. Michael A Cusumano, *"Envisioning the Future of India's Software Services Business"*, Communications of the ACM, Vol.49 No.10, October 2007
4. Infosys Annual Report 2008-2009, <http://www.infosys.com/investors/reports-filings/annual-report/annual/Documents/Infosys-AR-09.pdf>, p40
5. Kojima Makoto, "Indo no Software Sangyo" (Indian Software Industry), Toyo Keizai Shinpousha, 2004, p119-121
6. Kojima Makoto, "Indo no Software Sangyo" (Indian Software Industry), Toyo Keizai Shinpousha, 2004, p133-137
7. Infosys Annual Report 2008-2009, <http://www.infosys.com/investors/reports-filings/annual-report/annual/Documents/Infosys-AR-09.pdf>, p41
8. V. Subramanyam, Sambuddha Deb, Priya Krishnaswamy, Rituparna Ghosh, *"An integrated approach to software process improvement at Wipro Technologies: veloci-Q"*, Technical Report CMU/SEI-2004-TR00, p29-30
9. Steve Hamm, *"Bangalore Tiger: How Indian tech upstart Wipro is rewriting the rules of global competition"*, McGraw-Hill, 2006

## Chapter 6

1. Mitsuyuki Katakame, Soutaro Hase, "Saihen ni mukau jutakugata system kaihatsu industry", (Restructuring of Customize Software Business), Chitekishisansouzou Aug 2009, Nomura Research Institute Ltd, 2009, p56
2. Mitsuyuki Katakame, Soutaro Hase, "Saihen ni mukau jutakugata system kaihatsu industry", (Restructuring of Customize Software Business), Chitekishisansouzou Aug 2009, Nomura Research Institute Ltd, 2009, p55