Abstract

Under the keen competition of telecommunication market, to maintain and increase the market share, Com-S (The company I’m working now, hereafter refer as “Com-S”) needs to maintain the quality of service and to create more value-added services to customers. As more and more value-added services created, there is a need to update the existing applications and to develop new applications for both running the services and supporting the customer services.

Requirement collection is an important stage of application development and application update, with experiences gained in previous requirement collection stages, miscommunication problem among intra-business unit end users, third party stakeholders, System Development Unit colleagues is discovered to be the main issue that has greatly affected the quality of the system being developed and also the service level of Com-S.

In order to overcome this problem, the Use Case Model is recommended to apply in the requirement collection process, and three stages, namely “Analyzing the problem”, “Understanding the users and stakeholders need”, and “Developing the supplementary specification” are involved in implementation.

“Analyzing the problem” is focus on identifying problems, searching root causes, presenting the boundary of the suggested application being developed graphically, and listing out all constraints that the system is facing and their reasons behind.

Interviews are the main course of “Understanding the users and stakeholders need”. By interviewing users and stakeholders, our SDU colleagues can collect more clear and adequate requirements. Guidelines and
sample question sets will be created to ensure the quality of interviews. All collected information will then be sorted in accordance to the level of importance, under which decision will be made in corresponding to the requirement with higher priority.

“Developing the supplementary specification” is the stage of collecting non-functional requirements, which have not been collected in previous two stages. Non-functional requirements include the usability, reliability, performance, and supportability of the new or updated application. Design and layout requirements (if any) and the non-functional constraints are also being discussed in this stage. A supplementary document is created for recording all the requirements for further references.

Formal and instructive guidelines greatly help in ensuring the quality of each requirement collection process. And the implementation of the Use Case Model can lead to the creation of these guidelines which can be shared among SDU colleagues in order to enhance the overall quality of the new application development.
# Table of Contents

Abstract ...........................................................................................................................................2  
Introduction .....................................................................................................................................6  
   Company Business..................................................................................................................6  
   Company Background ..........................................................................................................7  
   Introduction of Business Units ..............................................................................................8  
      System Development Unit ..............................................................................................8  
      System Maintenance Unit ..............................................................................................8  
      Design Unit ........................................................................................................................9  
      Wholesale Unit .................................................................................................................9  
      Retailing Unit ....................................................................................................................9  
      Logistic Unit ......................................................................................................................9  
      Accounting Unit .............................................................................................................10  
      Customer Service Unit ..................................................................................................10  
Communication Channels Overview .........................................................................................11  
Mini-Cases .....................................................................................................................................14  
   #1. Miscommunications with end-users ............................................................................14  
   #2. Miscommunications with 3rd parties ............................................................................16  
Problem Statement ......................................................................................................................19  
Literature Review ..........................................................................................................................20  
   Definition and Essential Elements of Use Case .......................................................................20  
   Why Use Case .......................................................................................................................21  
   Use Case and Software Lifecycle - The Iterative Approach ..................................................22  
      The Waterfall Model (Dean & Don, 2003) ........................................................................22  
      The Spiral Model (Dean & Don, 2003) ............................................................................23  
      The Iterative Approach (Dean & Don, 2003) ....................................................................24  
3 requisite skills for effective requirements collection .................................................................27  
   Skill #1 - Analyzing the problem .........................................................................................28  
   Skill #2 - Understanding the users and stakeholders need .................................................31  
   Skill #3 - Developing the supplementary specification .......................................................34  
Brief Summary .................................................................................................................................38  
Recommendations ..........................................................................................................................41  
   Tools and actions for each step .............................................................................................41  
      Recommendation #1 - Full Approach ..............................................................................49  
      Recommendation #2 - Time Effective Approach ..........................................................51  
      Recommendation #3 - EDI Application Approach .......................................................53  
Summary of the recommendations .............................................................................................54
Introduction

Com-S is a small-medium size telecommunication company. When Com-S started its business 10 years ago, only a few staffs were employed and everyone was responsible to a specific business function.

Com-S has been expanding in the past 10 years, number of staffs were increased from less than 10 staffs in the beginning to around 30 staffs now. During this period, number of business units, products and services has also been increased too.

Company Business

In the very beginning, I would like to have a brief introduction of my company’s background and business. As mention above, Com-S is a SME telecommunication company. We offer customers SIM cards, which support both local and international call services, SMS, as well as other value added services, such as musical ringtone, miss call alert, etc. Yet, out of all services available, IDD services are the main focus in serving our target customers.

For the SIM card telecommunication services, as we are not strong in setting up telecommunication infrastructure, we’ve outsourced the setup processes to our partnering telecommunication provides, while at the same time concentrating in our strength, which is to provide value-added services, to retain existing customers, and to enlarge our customer base.

For the IDD services, we also have our contracted partners from various countries supporting us for both hardware and telecommunication services. While having partners to provide us with equipments for
setting up IDD services and to provide physical and virtual telecommunication connections for our IDD services, the frequency of communication has increased a lot when compared to having all services done in-house.

Company Background

After introducing the nature of our company business, I would like to further introduce some background of my company. When my company first starts as a small business with only a few staffs, there was no well defined organizational chart developed. As a result, everyone is responsible for more than one business function. Having run the business for ten years, our company has expanded with more than 30 staffs are being employed at the moment. Not till now, a more solid organizational structure has been identified but yet to be well defined by top management. Following (Exhibit 1) is the organizational chart merely defined according to my understanding of my company, where the size of each text box does not represent the size of the business unit.

Exhibit 1
Introduction of Business Units

In the following paragraphs, I will explain each business unit (from left to right) and the inter-relationships among them.

*System Development Unit.* The name of this unit is pretty self-explanatory that the responsibility of this unit is on system developments. The scope of system development includes developing in-house administrative systems, in-house operational systems, and value added service systems for SIM card users. By nature of development processes, communication with all other units seem to be inevitable for every system developers; and communications will be required in processes like system requirement collection and UAT, etc. Furthermore, since we co-operate with other telecommunication service providers in Hong Kong on SIM card services, this unit also needs to communicate with these third parties on defining the standard for data interchange and initiating complaints when any exception found after system production launched.

*System Maintenance Unit.* This unit is mainly focus on monitoring the connections between us and our worldwide telecommunication service providers. Apart from monitoring, colleagues under this unit are responsible in maintaining the in-house hardware and computer troubleshooting. On the other hand, this unit is closely co-operated with Wholesale Unit, Customer Service Unit, and System Development Unit. Problem raised by customers will be firstly received by Customer Service Unit or Wholesale Unit, according to the customer type, and then pass to System Maintenance Unit. System Maintenance Unit will then work with System Development Unit in analyzing and solving the problem. After the problem is
solved, the result will be pass back to the corresponding unit and the colleague there will call back to customer and do the follow up. This is the unit I’m working in now.

*Design Unit.* As you can recognize from its name, designers are the key members of this team, where they are responsible for designs like product package, promotion posters, banners, e-cards on Christmas, etc. Since they are to report our CEO directly, CEO becomes the person who needs to contact with the colleagues in this unit. Beside designing jobs, staffs under this unit will also need to support retail promotion events and marketing events, which leading to quite frequent communication with the colleagues in retail unit. For sure, they need to communicate with the technical team, especially the System Maintenance Unit, on any hardware problem.

*Wholesale Unit.* The main responsibility of this unit is to retain and to find more customers for our wholesale segment. The daily operations include communicating with wholesale customers, monitoring the traffic between we and our customers, making complaint when there is problem, and receiving complaint from our customers when our services have gone wrong. With the monitoring function, colleagues in this unit need to work closely with the System Maintenance Unit, including initialing ad-hoc connection testing, voice quality testing, etc. Furthermore, they have to report or request the colleagues in System Maintenance Unit to locate and solve any problem whenever complaints are received.

*Retailing Unit.* The function for this unit is more or less the same as the Wholesale Unit, but the target customers are those end-users instead of other third party telecommunication operators or service
providers. Although both Retailing Unit and Wholesale Unit need to retain and expand the IDD market share, Retailing Unit only needs to focus on Sales and Marketing of the retail segment, while leaving the quality monitoring and service assurance section to the System Maintenance Unit and all complaints from end-users to the Customer Service Unit. Retailing unit also work closely with Logistic Unit, which will be introduce in the next paragraph, since calling cards need to deliver to our retailing points in order to reach our end-users.

*Logistic Unit.* A Unit works closely with Retailing Unit and the main responsibility is to deliver our calling card to all the retailing points in-time and accurately. Since it works closely with Retailing Unit, communication between these two units are important, or else misunderstanding occurs will lead to delay in delivery, which in turn will affect the availability of the products in the market.

*Accounting Unit.* An important Unit inside Com-S as this is the unit who controls the financial status and cash flow of the company. Numerous of reports need to be generated in order to support the daily operations of this Unit. With this special nature, this unit needs to communicate with System Development Unit to have the appropriate reports. Moreover, adhoc information enquiries also require communication between these two units. Apart from System Development Unit, Accounting Unit also needs to work with Wholesale Unit and Retailing Unit for gathering the most up-to-date sales information and costing information.

*Customer Service Unit.* This is the frontline of Com-S and this is the first contact point to all end users. There are many communication channels for this unit, e.g. communication with retail customers, transfer
of technical issues from customers to the colleagues in System Maintenance Unit, communication with System Development Unit whenever there is a new need or enhancements for the Customer Service System, etc.

Communication Channels Overview

After a brief introduction of my company, it’s not difficult to picture that intercommunication is an important issue for the business. Below is a graphical presentation of the communication channels, as illustrated in exhibit 2 which is derived from exhibit 1, within Com-S and with Com-S’s outsiders.

![Exhibit 2](image)

Since all communication channels are interconnected, business unit as a whole will be affected if any miscommunication occurs in between. e.g. one of our mobile user calls customer service hotline, and complains about the malfunctioning of our SMS service, which requires our investigation into the
problem and need our investigation. If our customer service officer mis-communicates with our customer, the information gathered maybe insufficient or even wrong information is collected.

If insufficient information is passed to our System Maintenance Unit, all the following units will not be able to perform any follow-up procedures; as a result, our customer service officer has to re-contact customer again in order to collect all missing information. The time waste in information re-collection and re-interchange will increase the down time and customer has to suffer from this dispensable interruption of service.

If wrong information is collected and passed to System Maintenance Unit, although follow-up procedure can be carried out, the result is not what the customer expected. In turn, the whole problem investigating procedure has to start all over again in order to solve customer’s problem correctly.

Both of the above mentioned situations will not only affect the quality of our customer services, but also will lead to the possibility of losing customers to other service providers. In turn our market share will shrink, revenue income will decrease, and eventually the whole business will be affected.

Apart from information interchange from Customer Service Unit, both System Development Unit and System Maintenance Unit need to communicate with other units. These two units are the units that have the largest communication channels to both internal users and outsiders. If problems are found in these communication channels, the affection to the whole business will be severe. In the following section, I will use some mini-cases to illustrate some miscommunications between System Development Unit and other units or outsiders, which need further studies and some recommendations for overcome the
miscommunication problems.
Mini-Cases

#1. Miscommunications with end-users

This case happened after I had joined Com-S for around 3 months entities involved were highlighted in exhibit 3. A Customer Service Application was being used for several years in Com-S, its performance was good, yet modification or creation of new functions could hardly be made in order to meet the fast expansion of our business. Furthermore, the existing application was not a web based application, where individual installation would be necessary for every machine. When new colleagues joined Com-S, it would become very time consuming for all installation processes. The situation was even worse when upgrades were required for the application, where we would need to identify all machines and to do the upgrade one by one manually.

Under these circumstances, Com-S decided to re-engineer the Customer Service Application and migrate
Implementation of Use-Case model to solve the miscommunication problem in requirement collection process

it to a web based application and I was assigned to take over this re-engineering and migration project. Since I was new to the company and I was not familiar with the application, under the situation where system documentation was lacking, I would have to take application trials by myself in order to collect more information.

After several rounds of trials, I had collected some information about the application, but there were still many unclear points and variables which needed to be clarified. Aiming to know more about the requirements, user interface layouts, and expected outcomes for the application, I started interviewing end-users. Since they had been using the original Customer Service Application for a long time, they understood what was “right” and what was “wrong” to the application. As all existing users would avoid providing “wrong” information in order to get away from possible exceptions, I would suppose all the information collected from interviews was exception free information, and thus I had used those information collected from existing end-users for the re-engineering process.

As some information collected were not adequate, the result of the re-engineering was not satisfactory. Although the user interface was more or less the same as the original one, exceptional results were found, some outputs from the application were not as what the end-users expected. As a result, re-interviews were needed, and re-engineering was required to be done on some unsatisfactory functions. The process of re-interview and re-re-engineering process were a waste of time. Moreover as the new Customer Service Application was un-satisfactory, our regular customer service level was being affected and only limited services could be provided to customers.
#2. Miscommunications with 3rd parties

In order to maintain the current and to capture more market share, Com-S started a new business in mobile sector one year ago, which is my second year work in Com-S, and this new mobile service business was in partnership with another telecommunication company (hereafter refer as Partner C) in Hong Kong. Both Partner C and System Development Unit were highlighted in exhibit 4 for a graphical presentation on the relationship of the entities involved.

With different core competences, the partnership was established with Com-S focusing on billing, marketing, promotions, customer services, and add-on service innovation while Partner C would be focusing on the mobile service infrastructure and backend support. Because of the separation of infrastructure and billing ownership, asynchronous problems occurred, such as, asynchronous billing data, asynchronous service suspension updates, etc. In order to overcome these problems, data interchange
between Partner C and us was inevitable and this was the area where miscommunication occurred.

An agreed format standard was indispensible for data interchange, since Partner C already had more or less the same partnership with other telecommunication service providers, we had to follow their standard for the data interchange. An official documentation for data interchange was given by Partner C in the early stage of system development and meetings were held in different development stages. Although there were official documentations and meetings, some specific requirements were added and collected from Partner C. Our development on the data interchange ran smoothly with the information collected.

Several rounds of user acceptance test (UAT) were carried out before the launch of the mobile service, system was improved in each round and we believed that the production launch should be smooth and problem free. But when the service launched, we discovered that the actual number of data files being sent to us was abnormal, problem which was not being mentioned in any documents nor meetings. When in requirement collection, information collected as data file would be given in every 10 minutes. In UAT, only one set of data files was uploaded in each period, but more than one set of data files were uploaded in each period during the production stage. As a result, we encountered exceptions in our production environment and data was either not be able to process or being processed twice in each period.

It was lucky that the data volume in the initial production stage was small, and only a small number of customers was affected. Since all data collected was intended for billing, we had to take efforts in correcting the billing data and fine tune the programs in order to work in the production environment. We questioned Partner C on the reason for why such situation was neither mentioned in any documentation
nor in any meetings. They told us that it was their procedures to upload more than one set of data files in each period since they would capture data twice in their side within the agreed time period.

The miscommunication between Partner C and us was probably due to the fact that each party was accustomed to their own procedures, which were presumed to be universally understood by others. Due to miscommunication, we had wasted our time in re-engineering and also wasted our efforts in working on unnecessary and avoidable remedial measures.
Problem Statement

Miscommunication, in terms of inadequate, incomplete, and misleading requirement collection, occurs in Com-S, not only between business units inside Com-S but also with third parties. This problem led to a waste of time in collecting inadequate information, which in turns affecting the system development, and a waste of efforts in working on unnecessary and avoidable remedial measures. Moreover, the quality and stability of our services are being affected, customer loyalty is being affected, and our company reputation is being influenced as well. In order to maintain Com-S’s business and competitiveness, the miscommunication problem should be taken care of and solved eventually.

Will there be any possible managerial solutions for overcoming the miscommunication problem of Com-S to smooth the process of system development? In particular, special attention should be paid to the requirement collection stage of system development life cycle which seems to be the root of all problems.
Literature Review

In order to overcome the miscommunication problem inside Com-S, I’ve tried to search for methodology that is useful to this situation. At the end, I was recommended to have further studies in Use Case methodology and try to apply the theories and steps of Use Case for the solution. According to Kurt & Ian in 2002, use case is a powerful requirements-modeling technique, which provides a standard way of capturing, exploring and documenting what a system should do.

Definition and Essential Elements of Use Case

A use case can be defined as a sequence of actions a system performs, these actions are invoked when a particular actor, who is the external entity to the system, initiates the system and the actions will return an observable result of value to that particular actor (Dean & Don, 2003). Another definition of use case by Alistair in 1999 is, it is a description of the possible sequences of interactions between the system under discussion and its external actors, related to a particular goal.

From the definitions in above, both Dean & Don and A.Cockburn mentioned four essential elements of a use case. They are

- Sequence of actions. It describes a set of functions, algorithmic procedures or any other internal processes of a system. Results are produced via these actions. An action is atomic, i.e. performed either entirely or not at all. Furthermore atomicity requirement is greatly affected the selection of granularity of the use case.
System performs. It illustrates the functionality describe in use case. The works are triggered whenever an actor makes an order to the system.

An observable result of value. It is the core of the use case since use case must be “valuable” to users. This result must be meaningfully descriptive and motivate the users to interact with the system.

A particular actor. It is someone or something that the system being designed has to interact with (Pete, 1998). Only three types of users are considered as actor, they are

- Users: Human beings whom act on the system.
- Other systems or applications: Software which interacts with the system.
- A device: Both input and output devices are the interfaces to the system.

Why Use Case

As Dean & Don in 2003 cited that, use case is a methodology that tells a better requirement story, i.e. better understand by the prospective users (who use the system), the developers (who write functions and implement the system), the testers (who use use cases as a basis for testing) and the documentation team members (who write user guides and help). Furthermore, there are plentiful of benefit of using Use Case, such as

- Relatively easy to write and read when comparing with traditional requirement methods.
- Force developers to think and design the system in the users’ point of view.
- Engage users in requirement process, provide a way for them to communicate and document their
needs.

✓ In most circumstances, developers write the use cases, which make them not only understand

requirements but also responsible for determining them.

✓ A critical tool in design and implementation process, reduce the risk of implementing a “away from

requirements” system.

Use Case and Software Lifecycle – The Iterative Approach

Traditional effective requirements management can occur only when the context of a software process is
reasonably well-defined. This process defines a full set of activities that must execute in order to deliver
the final software product. There are different software processes for software development, e.g.
Prototyping, Top-down & Bottom-up Design, Waterfall Model, etc. Since the combination of the best of
waterfall model and spiral model is the iterative approach that Use Case used, a brief introduction of
waterfall model and spiral model will be illustrated in the following paragraphs.

The Waterfall Model (Dean & Don, 2003)

In the 1970, Royce defined the waterfall model, exhibit 5, of software development. Software activities
proceed logically through a sequence of steps. Each step bases its works on the activities of the previous
step, but no prototyping activity is prescribed. The waterfall model is successful in reinforcing the role of
requirements, requirement analysis is the very first step of the model and serves as the basis for design,
coding and testing activities which logically occur after requirements.
This strength eventually becomes the primary source of weakness of the waterfall model, requirements are being “frozen” in the first stage of the development and change of requirements is anathema. As a result, the development may become completely disengaged from the real world. With an inadequate understanding and out of date of the requirements, the final product may be non-deliverable, have chaos, poor quality and out of user expectations.

*The Spiral Model (Dean & Don, 2003)*

Structured with a waterfall-like process, the spiral model (exhibit 6), which is introduced by Boehm in 1988, is also a step-by-step model. Development is initially driven by a series of risk-driver prototype; final system will be produced stage by stage like the waterfall model.
Unlike the waterfall model, the spiral model provides a sensible road map that helps to address some requirement changes and take them into account. The spiral model starts with requirement planning and concept validation, followed by numbers of prototypes to assist the understanding of the system requirements in the early stage. The benefit of this model is the availability of multiple feedback opportunities with the users and customers, but the quality of the feedback is poor, users always intended to provide something new in any feedback opportunity which makes the process slows down. Furthermore, developers typically do not have the time for full concept validation and numerous prototype developments, followed by a rigorous waterfall methodology.

*The Iterative Approach (Dean & Don, 2003)*

The Iterative Approach, introduced by Kruchten in 1995, of Use Case used is the combination the best of...
The Waterfall Model and The Spiral Model. This approach has proved effective in a wide variety of project types and number of advantages over the waterfall models and spiral models.

In iterative approach, the lifecycle phases are decoupled from the logical activities, allowing revisit of various activities, such as requirement analysis, design and implementation, during various iterations of the project. In addition, each iteration is designed to alleviate the risks which maybe occur in that particular stage of the development activities.

The iterative approach consists of four lifecycle phases, they are Inception, Elaboration, Construction and Transition. These phases are corresponding to each state of the project at different time. A brief description of the four phases will be illustrated in exhibit 7.

- Inception Phase. The beginning phase of iterative approach, it focuses on understanding the project. Understanding areas include business cases, scope of the project, and the feasibility of the implementation. Unlike the waterfall model, requirements are not collected in this stage but problem analysis is performed, the vision for the solution is created. Preliminary schedule, budget and risk factors are defined in this phase too.

- Elaboration Phase. The second phase follows by inception phase. System requirements are defined here. Initial architecture is established. Early feasibility prototype is developed and demonstrated in this phase.
• Construction Phase. The third phase of iterative approach and focus on implementation. Most of coding occurs in this phase, architecture and design are fully developed. The time acquired in this phase is longer than the previous two phases.

• Transition Phase. Beta tests, user and maintainer trainings occur in this transition phase. The system is transitioned to user and deployed for use.

Multiple iterations, illustrate in exhibit 8, occur within each phase of the iterative approach. Sequences of activities with established plans and evaluation criteria, resulting in an executable of some type is the definition of “iteration”. Sequences of activities of iteration are built bases on the activities of the prior iteration, accumulating these iterations will result in “iterative and incremental” software development fashion.

Different iterations are selected according to the number of criteria. Studies for the viability of the chosen architecture against the project should be designed in the early iterations. If the chosen architecture is not feasible, a change of the architecture must be consider before any development begins.

A set of disciplines is organized in every activities associated with the development in the iterative approach. Discipline is a logically related set of activities, and discipline defines how the activities must
be done in sequence in order to give a viable work product. During each iteration, adequate time must be spent on the discipline of each phase. Iterations can be regarded as mini-waterfalls through the activities of requirements, analysis and design, and so on, but is tuned to specific needs of different iteration. For example, adequate time should be spent on problem analysis and feasibility studies in inception phase while significant time should be spent on “refining” the requirements in elaboration phase.

3 requisite skills for effective requirements collection

According to Dean & Don in 2003, 6 skills are required for effective requirement management. Since this project is focusing on a solution for requirement collection miscommunication problem, and 3 out of these 6 skills are related to requirement collection, only 3 skills will be mentioned. These three skills namely

- Analyzing the problem. Focus on develop a set of techniques to gain a proper understanding of the problem.

- Understanding the users and stakeholders need. Techniques are acquired to better understanding the actual needs of a user or stakeholder and the real needs that the system must address.

- Developing the supplementary specification. Since many kinds of functional requirements cannot conveniently express from the second skill, especially those non-functional and design requirements, this skill is used for catching these non-captured requirements.
Skill #1 – Analyzing the problem

Since not every application is developed to solve a problem, e.g. some are built due to match with the changes of market; some are built to take the advantage over competitors; even when the existence of the problem is not clear. Therefore we need a definition on the phrase “Problem” and “Problem Analysis” before having further illustration.

Problem is defined as “the difference between things as perceived and things as desired” (Dean & Don, 2003), i.e. there is a gap between what users already have and what they desired to have. Base on this description, problem analysis is defined as “the process of understanding real-world problems and need of the users and proposing solutions to meet their needs, or minimizing the gap between what users have and what they desired to have”. As a result, the goal of problem analysis is to gain a better understanding of the problem begin solved in users’ point of view before development begins.

In order to achieve the goal of problem analysis, the following five steps must be taken.

Step 1. Gain agreement on the problem definition.

One of the ways to gain this agreement is to write the problem down and find for users’ and stakeholders’ agreement. With this agreement, we can make certain that the benefits are described in the terms of users’ point of view. Thus gain a better understanding of the users’ and stakeholders’ view of the problem.

During the gain agreement exercise, users and stakeholders have the opportunity to provide reorientation which may trigger the development of different system architecture.
Step 2. Understand the root cause

A systematic way to understand the uncovered root cause of a problem is applying root cause analysis. Fishbone diagram, exhibit 9, is a technique used in root cause analysis. Every root cause discovered is listed as one of the “bones” inside fishbone diagram, each bone is prioritize and consider whether the source worth fixing or not by determine the materiality of each root cause.

Step 3. Identify the users and stakeholders

Stakeholders can be categorized into direct and indirect users, both of them may have influences, either trivial or nontrivial, to the system. Therefore understanding of stakeholders and their particular needs is important. It’s a good practice to have a summary of users and stakeholders identification for future references.

Step 4. Define the solution system boundary

System boundary, exhibit 10, is used to define the system that can be deployed to the addressed problem, i.e. an envelope in which the solution system is contained. In doing so, two things have to keep in mind: understand the problem and consider a potential solution.
Step 5. Identify the constraints to be imposed on the solution

Constraints must be considered in analysis since they have the potential to severely restrict the ability to deliver a viable solution. The potential sources of system constraint are Economics, Politics, Technology, Existing Systems, Environment and Schedule & Resources. Elicit the majority of the constraints that affect the solution and change them to becomes requirements for the new system and overcome them.

Business modeling is a problem analysis technique suitable for IT environment (Dean & Don, 2003). The purpose of this modeling technique is to understand the structure and dynamics of the existing organization, ensure developers and users have a common understanding and how the new system facilitate productivity and affect the existing systems. To drive application development under business modeling, two key models are used, they are business use-case model and business object model.

Business use-case model, exhibit 11, is an essential input to identify roles and deliverables in the organization. It consists of actors (something interacts with system) and the use cases (sequences of events to get jobs
done). With both actors and use cases, business use-case model describe who is involved in the business activities and how these activities take place.

Business object model, exhibit 12, describes the entities and how they interact to deliver the functionality necessary to realize the business use cases. It shows how the business use cases are performed in terms of interacting business workers and business entities.

With both business use-case model and business object model, a comprehensive overview of how business works is provided. In addition with the five steps describe before, we can have a better understanding of the problem in users point of view, which affect the quality of problem analysis.

Skill #2 – Understanding the users and stakeholders need

Users and developers are typically from different worlds, have different backgrounds, knowledge and experiences. Differences on point of views and understandings between users and developers are always the issues, thus the real-world problems are not fully understandable by developers.
In order to minimize the gap between users and developers, a variety of techniques can be used to gain better understanding of real user and stakeholders needs, e.g. interviewing users, data collection using questionnaires, requirement workshops, brainstorming, etc. Which one or what combination of the techniques, is the best depends on the situations, such as

- Types of application being developed
- Skills & sophistication of developers
- Skills & sophistication of users and stakeholders
- Scale of problem

Interviewing is a simple and direct technique that can be used in most circumstances. One of the key goals of interviewing is to make sure the biases and predispositions of the interviewer do not interfere with a free exchange of information. To avoid interference, questions asked via interviews should be context-free. Context-free question means a question asking for the nature of problem without any context for a potential solution, e.g. Who is the user? Are their needs different? A context-free question helps to gain a real understanding of the problem in focus without biasing the user’s input by developer’s “expert knowledge”.

After context-free questions have been asked and answered, it may be appropriate to move questions to a point that helps to explore for solutions. These kinds of questions are named as solution-context questions. e.g. What are the expectations? How do you currently solve this problem? After all, not only developers can understand the real problem and can provide appropriate solution to the problem, but also may give
the user new insights and perhaps a different view of the problem which helps to have a better solution.

With some preparations and a structured interview template (a list and sequence of questions need to ask during interview), an adequate job of interviewing should be able to carry out. In order to make the interviews even more success, Dean & Don in 2003 suggest the following tips,

- Jot down the context-free questions in a notebook for referencing. Review the questions before interviews
- Research the user and stakeholder’s background for some understandings. On the other hand, don’t try to briefly verify the answers with the interviewee.
- Jot down the answers in a notebook instead of in any electronic way.
- Refer to the template during interview and make certain asking the right question

Apart from question design, another important issue for interviews is to prioritize the requirements. Since no development team can response to all requirements due to the scope and time limit of the development project, some measurements on the requirements are needed for prioritize their importance. An effective prioritization technique is “Critical, Important & Useful” categorization and the definition, by Dean & Don in 2003, of the three categories are

- Critical means indispensable, suggesting that the system will not viable if such requirements are not fulfilled.
- Important means that the requirements can be put aside in the meantime but need to fulfill in the coming versions.
Useful means nice to fulfill such requirements, which make the system to become more user-friendly and operation will become simpler and easier.

In this technique, each participant is given a number of votes equal to the number of requirements, but each vote must be either “critical”, “important” or “useful”. In order to avoid the deflection to “all critical” situation, each participant is given only one-third of the votes from each category, thus only one-third of requirements can be considered as critical. Furthermore, in order to avoid the mix of categories, especially when prioritize by a large number of users, we can give a weighting to each category, e.g. “critical” weight as 9, “important” weight as 3 and “useful” weight as 1, and simply add up the weightings for each requirement. The result will heavily favor the “critical” votes and thus the really critical requirements will not be easily hide up.

Skill #3 – Developing the supplementary specification

After completing the above two skills, around 80% of the functional requirements, defined as service, tasks or functions the system is required to perform (Ruth & Dana, 1999), should be captured in use-case form and understand by developers. However effective requirement collection should include those 20% non-captured requirements, e.g. primarily algorithmic and computational in nature requirements, embedded-controls system requirements, etc, otherwise requirements are not fully collected and developed system will be non-viable. To assist with reasoning, discovery and completeness, these non-functional requirements is organized into four categories (Dean & Don, 2003): usability, reliability, performance and supportability.
Usability means ease to use. In today’s software products, usability rank as one of the top criteria for success. However this tends to be subjective to the beholder, i.e. users and stakeholders, there is no simple solution to this problem, the best way to cope with this is to offer a set of guidelines to help in addressing the usability requirements. Some suggestions follow.

- Specify the required training time for a user to become minimally productive and operationally productive. This may need to be further described in terms of novice users, normal users and power users.
- Specify measurable task time for typical tasks or transactions that the end user will be carrying out. Task time includes response time provided by the system and the task performance time, these two times should be specifying separately to have a clear cut-off.
- Compare the usability of the new system with other state-of-the-art systems that the user and stakeholders knows and likes.
- Specify the existence and required features of online help systems, wizard, tips and other forms of documentation and assistance.

Reliability describes the degree to which the system must behave in a user-acceptable fashion. No matter how optimistic a user is, he or she will aware that things do go wrong. With the reliability requirements, we are able to understand the minimal acceptance of the system. Typically reliability includes the following issues.

- Availability. This describes how long the system must be available for operational use during a
certain period of time. E.g. 7x24 means a non-stoppable system, 99% available in office hours, etc.

- Mean time between failures (MTBF). This describes the average time the system is allowed between each failure. Usually specified in hours, but it also can be specified in days, months or minutes depends on the system availability requirement. A precise meaning of “failure” must be defined.

- Mean time to repair (MTTR). This describes the maximum time the system is allowed to be out of operation after failure. A range of MTTR values maybe appropriate due to different scenarios, e.g. 90% of all system failure must be repaired within 5 minutes while the remaining 10% must be repaired within 30 minutes. Like MTBF, a precise meaning of “repair” must be clarified.

- Accuracy. This is the precision requirements for the system. E.g. the rounding of financial report should be up to the nearest cents or dollars.

- Maximum bugs or defect rate. Express in terms of bugs/KLOC (thousands Lines Of Code) or bugs per function-point.

- Bugs per type. Express the number of bugs allowed according to their severity, e.g. severity category can be “minor”, “significant” and “critical”. Definitions for each category are important too.

Performance is self-explanatory. This requirement usually covers several categories of a system, such as

- The average or maximum response time of a transaction.
Implementation of Use-Case model to solve the miscommunication problem in requirement collection process

- Throughput which means number of transactions to be done per a specific time frame.
- Capacity. It’s the number of users or transactions the system can accommodate.
- Degradation modes. This describes the acceptable mode of operation when the system has been degraded.

Supportability defines the ability the system to be easily modified to accommodate enhancements and repairs in the future. The supportability requirement can stipulate different response times the maintenance group under different enhancements. Enhancements can be further categorized into simple enhancements, moderate enhancements and complex enhancement. Once again, precise meaning for simple enhancements, moderate enhancements and complex enhancements must be clearly defined.

In addition to the non-functional requirements, design requirements collections can help in understanding the design constraints. The word “design constraints” is defined as some restrictions on the design of a system or the process of a system that do not affect the external behavior of the system but that must be fulfilled to meet technical, business, or contractual obligations. Typically there are three sources of design constraints, they are

- Restriction of design options. E.g. the RDBMS can be use, development tools can be use, etc
- Conditions imposed on the development process. This kind of constraints is most likely to be found when specifying the team’s development infrastructure, e.g. compatibility to existing systems, use of specific application standards, etc.
- Regulations and imposed standards. These are the requirements from the indirect users, e.g.
To cope with these design requirements, the best way to handle them is to follow some guidelines.

- Distinguish them from the other requirements.
- Include all design constraints in a special section, or use a special attribute so they can be readily aggregated, so that these requirements are easily to find and review.
- Identify the source of each design constraint for references.
- Document the rationale for each design constraint. This will help to become a reminder for the design constraint.

Lastly is the identification of other requirements. These requirements maybe come from a variety of sources, maybe similar to some categories listed in above, but still worth to recognize them and record them accordingly. These requirements may have impact on the system design and system development.

After the collection of the above non-functional requirements, design requirements and other requirements, document them in a supplementary specification for further references and system development.

**Brief Summary**

Skills and steps for requirement collection improvement of Com-S are being listed and discussed. A summary of these required skills and steps can be concluded as followings.

In the very beginning, we need to understand what the problem is in every application development. To
achieve this understanding, 5 steps are recommended by Dean & Don in 2003. They are “Gain agreement on the problem definition with users and stakeholders”, “Understand the root causes”, “Identify the stakeholders and users”, “Define the solution system boundary” and “Identify the constraints to be imposed on the solution”. In this stage, we can make use of Business use-case model and Business Object model to help in analyzing the problem.

After the stage of problem identification, we need to understand the needs of users and stakeholders. Interviewing is the most frequent and effective way to understand the needs. Before interviewing the users and stakeholders, a set of context-free and solution-context questions can be prepared which helps to ensure the quality of the interview and the information collected. After interviewing the appropriated users and stakeholders, we need to prioritize the collected requirements and assign to different development stages.

Most of the functional requirements are captured and documented in the above stage but not for non-functional requirements, therefore we need to develop supplementary documentations for these requirements. Non-functional requirements are categorized into usability, reliability, performance and supportability, which assist in identify the constraints of each function. Furthermore, design and other requirements need to be documented in the supplementary documentations too.

With the above three stages, I’m going to decide a solution for the problem of miscommunications in application requirement collection period. Since intra-business unit end-users and third party stakeholders have different natures, e.g. user availability, number of interviews taken, knowledge and understanding to
the applications, etc, a single one solution seems not suitable for solving all the problems.

Although intra-business unit end-users are available for interviews at anytime, they don’t understand technical issue, where sometimes they will only have the general idea of what they want but not the details of what they really needs. The solution for this kind of users includes analyzing the problem, understanding the needs of users, and making supplementary documents for those non-functional, design and other requirements.

Beside departmental users, interviews with third party stakeholders are also necessary though they might not be available for interviews often. These stakeholders are technical people who understand technical issues. By interviewing them, although problem can be well defined with appropriate solution, tailor make adjustment is necessary in order to have the solution fit into our system for electronic data interchange purposes. As a result, the solution is a little bit different from the one proposed for dealing with intra-business unit end-users. The mainframe is still a composition of analyzing the problem, understanding the needs of users and stakeholders, and making supplementary documents, and the only difference will be that not all discussed details are to be used in this stage.
Recommendations

Tools and actions for each step

Before any recommendation for solving the miscommunication problem is made, I would like to list out some tools and/or required actions which help in facilitate the rollout of each step in different stage.

- Gain Agreement on the problem with users and stakeholders

A “Problem Statement Form” is created for this step. (Exhibit 13).

![Exhibit 13](image)

The four elements in this form can help in describing the problem faced by the end-users.

- “Problem” is for stating a brief description of the problem.
- “Affect Parties” is for listing the stakeholders, who will be the target end-users which need to gain agreement with.
- “Impact” is for stating the effect of the problem on end-users and business activities.
- “Benefits of a solution” is for listing the benefits that the proposed solution can bring.

The information stated will keep changing and updating before any agreement is gained from the end-users.
• Understand the root causes

Fishbone diagram is used to identify the root cause of the problem, which is defined in “Problem Statement Form”. (Exhibit 14)

“Problem” is the “Problem” element described in “Problem Statement Form”, all the related causes will be shown and described in “Cause & Details”. When the number of causes listed is more than one, a “Cause Prioritization List” is required to prioritize the importance (start from 1 and ends with the maximum number of causes) of the cause and consider whether it worth fixing or not, and provide appropriate reason to support the consideration. (Exhibit 15)

• Identify the stakeholders and users

A “User & Stakeholder List” is used in this stage. (Exhibit 16).
The usage of this list is to identify all related users (direct user) and stakeholders (indirect user) of the system. The column “Comments” is used for drop down a brief notes which is useful for identifying the non-functional requirements in the successive step.

- Define the solution system boundary

In this step, both Business use case model and Business Object model will be used for drawing the block diagram of the system with related users and stakeholders. The number of block diagrams required depends on how sophisticate the system is, how many subsystems need to breakdown, and how many users and stakeholders are related to the system.

- Identify the constraints to be imposed on the solution

To facilitate the identification of constraints for the system, a “Constraint List” is created for listing the constraints and the rationale. (Exhibit 17)

<table>
<thead>
<tr>
<th>Constraint List for</th>
<th>System / Function #</th>
<th>Constraint Description</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Below is the descriptions for the columns,

- “System / Function #” is the assigned number for a specific system or function.
- “Constraint Description” is used for writing the description of the constraint.
- “Rationale” is used to write down the reason of the constraint.
• Interview – Guidelines

The quality of interview varies with different interviewer and the person being interviewed, hence it’s hard to guarantee that every interview can result with the same quality. In order to minimize the opportunity of conducting a worse quality interview, a set of guidelines is created for interviewers.

➢ Before interview

✓ Well prepared before conducting any interview, try to consolidate some context-free questions and drop them down in a notebook for reference during interview.

✓ Review those context-free questions before interview.

✓ Try to collect some background information of the person being interviewed, i.e. user and/or stakeholder.

➢ During interview

✓ Try not to bore the person being interviewed with question that can be answered in advance.

✓ Try to clarify and verify any unclear answers with the interviewee.

✓ Recommended to jot down the answers in a notebook instead of in any electronic form.

✓ Don’t cut the interviewee’s dialogue with another question when he/she is going to describe his/her bad feeling of the current situation.

✓ Ask follow-up questions about the answers the interviewee finish his/her answer.

✓ Refer to the template during interview to make sure that you are asking the right question.
After interview

- Summarize the gathered information.
- Prioritize the requirements collected by the nature of importance. Recommended to divide the importance into three categories, “Critical, Importance & Useful” and let the users and/or stakeholders to prioritize them.

- Interview – Sample Context-Free Questions

In order to avoid prejudicing the interviewee’s answers, a set of context-free questions should be used in interviews. Different set of questions should be used in different interview and each set of questions should be tailor-make for different requirement collection process. Below is a recommended sample set (Dean & Don, 2003 & Context Free Questions Samples) of context-free questions with its related section. (Exhibit 18)

Exhibit 18

List of sample context-free questions

<table>
<thead>
<tr>
<th>Section I: Creating user profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
</tr>
<tr>
<td>Position/Job title:</td>
</tr>
<tr>
<td>Key Responsibilities:</td>
</tr>
<tr>
<td>What kind of outputs produce?</td>
</tr>
<tr>
<td>How to measure the level of success?</td>
</tr>
<tr>
<td>What problem interfere the success?</td>
</tr>
</tbody>
</table>
List of sample context-free questions (Cont’d)

<table>
<thead>
<tr>
<th>Section I: Creating user profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anything can help to smooth the current processes?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Section II: Assessing the problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Which problem lacks solution / need improvement?</td>
</tr>
</tbody>
</table>

For each problem mentioned, ask follow-up questions, such as:

- Why does the problem exist?
- How to work with the problem right now?
- Any practices to cope with the problem?
- Any un-predicted problem/interference found when applying the practices?

<table>
<thead>
<tr>
<th>Section III: Understand the user / current environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is user’s education background?</td>
</tr>
<tr>
<td>What is user’s computer knowledge?</td>
</tr>
<tr>
<td>Is user experienced with the requested application?</td>
</tr>
<tr>
<td>What kind of OS / platform user is familiar with? Using what platform right now?</td>
</tr>
<tr>
<td>Going to change platform in the future? What platform? When?</td>
</tr>
<tr>
<td>Any expectation on training time?</td>
</tr>
<tr>
<td>What kind of user help expected? HardCopy / SoftCopy / Supportive Team</td>
</tr>
</tbody>
</table>
List of sample context-free questions (Cont’d)

Section IV: Before end

- Is there anything you would like to ask me?
- May I come back or call/email you with more questions later, if needed?

- Interview – “Critical, Important & Useful” categorization

To avoid deflection in users and stakeholders prioritization in requirements, this technique is used for categorization. Each participant is given only one-third of the votes to each category, in other words, only one-third of the requirements can be categorized as “Critical” while another one-third as “Important” and the remaining one-third as “Useful”. Below is the “Requirement Prioritization Form” for users and stakeholders to fill in the prioritization. (Exhibit 19)

![Exhibit 19](image)

<table>
<thead>
<tr>
<th>Requirement #</th>
<th>Requirement Description</th>
<th>Critical/Importance/Useful</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Developing supplementary documentation

Apart from collecting functional requirements, non-functional requirements are also need to document well in order to have a full picture of requirements for the application. Like those context-free questions listed above, different non-functional questions should be asked according to the nature of the target application. Following are some samples (Context Free Questions Samples) for developing the supplementary documentation. (Exhibit 20)
**List of sample questions for supplementary documentation**

<table>
<thead>
<tr>
<th>Non-functional requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System Usability</strong></td>
</tr>
<tr>
<td>◦ How long should a training course needed? For novice / normal / power users.</td>
</tr>
<tr>
<td>◦ Maximum time for a task to finish?</td>
</tr>
<tr>
<td><strong>System Reliability</strong></td>
</tr>
<tr>
<td>◦ What is the expected reliability? 7x24 / 0900-1700, etc</td>
</tr>
<tr>
<td>◦ The expected MTBF of the system?</td>
</tr>
<tr>
<td>◦ The expected MTTR of the system?</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
</tr>
<tr>
<td>◦ The acceptable response time for the system to response?</td>
</tr>
<tr>
<td>◦ Maximum time allowed for a transaction to carry out?</td>
</tr>
<tr>
<td>◦ The expected Capacity of the system?</td>
</tr>
<tr>
<td>◦ What is the expected performance?</td>
</tr>
<tr>
<td><strong>Supportability</strong></td>
</tr>
<tr>
<td>◦ Maximum time allowed for system enhancement?</td>
</tr>
<tr>
<td>◦ Any compatibility requirement?</td>
</tr>
</tbody>
</table>
List of sample questions for supplementary documentation (Cont’d)

<table>
<thead>
<tr>
<th>Non-functional requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Design</td>
</tr>
<tr>
<td>◦ Any preference on the layout design?</td>
</tr>
<tr>
<td>◦ Will the color of notification words different with different message type?</td>
</tr>
<tr>
<td>E.g. Alert Messages, Informative Messages, Error Messages, etc.</td>
</tr>
<tr>
<td>• Others</td>
</tr>
<tr>
<td>◦ Any regulations need to meet?</td>
</tr>
<tr>
<td>◦ Any restrictions on programming language can be used?</td>
</tr>
<tr>
<td>• Any restrictions on RDBMS?</td>
</tr>
</tbody>
</table>

Recommendation #1 – Full Approach

The first recommendation for solving the miscommunication problem is to go through all three stages and steps when collecting requirements. This approach is suitable for any application development when there is sufficient development time and resources, and users are always available to attend any requirement collection event.

Whenever there is a request for new application or for an application update, the stage of “Analyzing the problem” starts. Use the “Problem Statement Form” to write down the problem and try to gain agreement on the problem from the user and the stakeholder. This step is iterated until an agreement of problem is reached. Next step is “Understand the root cause”. “Fishbone diagram” is used to list all detail causes of
the agreed problem in previous step, and to evaluate the value of fixing these causes.

What follows is the step of “Identify the users and stakeholders”. Use “User & Stakeholder List” to list out all the direct and indirect users who may be related to the application for future reference and for identifying some non-functional requirements. With the problem agreed, root causes identified, and related users or stakeholders, we can “Define the solution system boundary” by drawing a substance diagram with boundaries using business use-case model and business object model. The last step of this stage is “Identify the constraints to be imposed on the solution”. In this step, a “Constraint List” is used to list all the constraints, its description and reasons for the constraints occur. After carrying out this stage, all the function requirements as well as all user and stakeholder’s information and system boundary should be collected.

The consequent stage after “Analyzing the problem” is “Understanding the users and stakeholders need”. The main action in this stage is to interview the users and stakeholders and try to understand what they really need. A set of guidelines for interview is created, which every interviewers need to follow. In short, interviewers need to be well prepared before any interview, e.g. prepare the context-free questions, collect some background information of the interviewee, etc. During the interview, interviewers need to drop down everything as fast and as details as they can, furthermore, they are required to ask follow-up questions and clarify the unclear points with the interviewees.

After each interview, interviewers are required to summarize all the requirements collected, and arrange for a requirement prioritization event with the users and stakeholders to list the importance of each
requirement. In order to avoid deflection on the importance of the requirements, a “Critical, Important & Useful” categorization technique is recommended for prioritization. Users and stakeholders are required to fill in the “Requirement Prioritization Form” for requirement prioritization.

The last stage of this recommendation is “Developing the supplementary specification”, the collection of non-functional requirements, around 20% of the overall requirements, is the target of this stage. A set of questions are recommended to ask according to system usability, reliability, performance, supportability and design in this stage. More questions are needed with different interviews and the nature of the new system.

Recommendation #2 – Time Effective Approach

Since not all the application development have sufficient time frame for requirement collection, another approach recommended is to minimize the time for requirement collection while keeping the minimal quality of the information collected.

The very first stage of requirement collection is still the “Analyzing the problem”. In this approach, the first step is to “Gain agreement on the problem with users and stakeholders”, the “Problem Statement Form” is still use for jotting down the problem and try to gain the agreement on the problem with user and stakeholder. Next is the step of “Understand the root cause”, it is also recommended to use “fishbone diagram” as in recommendation #1 to list all the causes and the details of the agreed problem. Evaluation on the causes is needed in this step to find out which cause worth fixing and which is not.
Unlike recommendation #1, the third step is skipped since those indirect users will have little influence on the application and those users & stakeholders whom agreed the problem will be treated as the direct users; thus in this approach, there will be a direct go to step 4 “Define the solution system boundary”. Business use-case model and business object model are still the models used in this step for limiting the scope of the application. Numbers of block diagrams will be drawn in order to clearly clarify the boundary of the application and each function. The first stage “Analyzing the problem” ends here and stage two starts.

Stage two is to “Understand the users and stakeholders need”. Interviews are arranged in this stage and interviewers also need to follows the “Interviewer Guidelines” as listed above. Interviewers still have to be well-prepared before interviews, a set of context-free questions are still needed but collecting interviewee’s background information is skipped, we can directly ask the person being interviewed and drop down all the information and requirements during the interview for reference.

After interview, it’s also recommended to summarize all the requirements collected. A requirement prioritization section is also arranged to the users and stakeholders to prioritize the importance of the requirements. In order to avoid category mix-up (Critical, Importance & Useful), it is recommended to give weighting to each category and try to increase the distance between each category. The “Requirement Prioritization Form” is also used and fills by the users and stakeholders for requirement prioritization.

“Developing the supplementary specification” is the last stage of this approach. Unlike in
recommendation #1, some non-functional requirements, e.g., design layout, online help availability, can be filtered; answers for these requirements will either have a default value or the current standard is to followed which is to be verified by users and stakeholders.

Recommendation #3 – EDI Application Approach

Problem and list of users and stakeholders are well defined before any application development, this is always the case of developing non-human interactive applications, but there is still an opportunity of collecting insufficient requirements, especially those non-functional requirements. The full approach seems can be used for solving this problem but it’s not the best solution. As a result, the third solution, System Interactive Application Approach” is recommended.

Since the problem, users and stakeholders are clearly identified, there is no need to start the “Analyzing the problem” from step 1, we can start the process from step 4, “Define the solution system boundary”. Although problem is clearly identified, the boundary of the system is not well defined. As a result, like in recommendation #1, it is recommended to draw the system diagram with business use-case model and business object model. With a clearly defined system boundary, we use “Constraint List” to list the constraints, description and reasons of the constraints in the consequent step, “Identify the constraints to be imposed on the solution”.

Next is the stage of “Understanding the users and stakeholders need”. Since we are focusing on developing System Interactive Application, no users can be defined but stakeholders still exist in this
situation. As a result, there is still a need for preparing context-free questions set. Interviewers also have to follow the guidelines, to be well prepared before any interviews or meetings, yet they can exclude the preparation steps for understanding the background of the stakeholders since this information is not relevant to the system.

Lastly it comes to the third stage, “Developing supplementary documentation”. This stage is more or less the same as the one recommended in recommendation #2, a set of questions are recommended to ask according to system usability, reliability, performance, supportability and design in this stage. Layout design and online help once again are irrelevant to the system development so these types of questions can be skipped.

Summary of the recommendations

After the explanation on the three recommendations, it seems that many steps are required for different approach. Below is a table summarizing the steps to be done for each recommendation.

Recommendation Summary

<table>
<thead>
<tr>
<th>Steps</th>
<th>Recommendation</th>
<th>Full Approach</th>
<th>Time Effective Approach</th>
<th>EDI Application Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 1: Analyzing the problem</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Gain Agreement on the problem with users and stakeholders</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Recommendation Summary (Cont’d)

<table>
<thead>
<tr>
<th>Steps</th>
<th>Recommendation</th>
<th>Full Approach</th>
<th>Time Effective Approach</th>
<th>EDI Application Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Understand the root causes</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>- Identify the stakeholders and users</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Define the solution system boundary</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>- Identify the constraints to be imposed on the solution</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

Stage 2: Understanding the users and stakeholders need

- Interview Guidelines                       |                                                                               | ✓             | ✓                       | ✓                        |
- Sample Context-Free Questions              |                                                                               | ✓             | ✓                       | ✓                        |
- “Critical, Important & Useful” categorization |                                                                               | ✓             | ✓                       |                          |

Stage 3: Developing supplementary documentation

- Developing supplementary documentation    |                                                                               | ✓             | ✓                       | ✓                        |

Evaluation of the recommendations

It’s impossible to apply all three recommendations whenever there is new application or application update requirement, so some evaluations are needed in order to find out the most suitable one. In this section, I’m going to evaluate these three recommendations according to the following aspects and I’ll
going to explain why I choose these aspects.

List of evaluation aspects and reasons

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Reason for evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step Completeness</td>
<td>As mentioned in the literature review, all steps are used for improving the quality of requirement collection. The more the steps to be implemented, the more accurate and informative requirements can be collected. Therefore the number of steps being implemented is affecting the output of the implementation.</td>
</tr>
<tr>
<td>Time Effectiveness</td>
<td>Time is money, the less time used in pre-development process, the earlier the development can be finished and implemented into our service level. As a result, the shorter the time used is preferred.</td>
</tr>
<tr>
<td>Human Resources Effectiveness</td>
<td>Implementation of new procedure requires additional input of human resources since more steps / rules have to follows. In the point of view of a company, more human resources input equals to higher cost for implementation. It is ideal that the current human resources can fit the implementation requirement which implies no more human resources is required.</td>
</tr>
</tbody>
</table>
List of evaluation aspects and reasons (Cont’d)

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Reason for evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost Effectiveness</td>
<td>Cost of implementation is the most important aspect for Com-S to accept the implementation or not. For a SME, cost control is important and investment on new procedure / practice is risky. The lower the cost, the higher the opportunity to accept the implementation.</td>
</tr>
<tr>
<td>Case Suitability</td>
<td>The problem identified is due to the incident created in the 2 mini cases. Since problems are reiterated, the higher relevancy of the recommendations to the cases, the higher the chance to have potential problem solved in the future.</td>
</tr>
<tr>
<td>Ready For Implement</td>
<td>Implementation of new procedure is both time and resources consuming. If resources are ready, implementation of the recommendations can be quick, otherwise more resources will be needed to input for the implementation.</td>
</tr>
</tbody>
</table>

Since different company will have different consideration on the above 6 aspects, the evaluation I’m going to make right now depends on the situation and cases of my company. A 5-point scale will be used for the evaluation, 0 means un-favor to the cases while 5 means favor to the cases. A spider-web chart will be used to represent the evaluation result.
5 points will be given to Full Approach in the aspect of “Step Completeness” since all the steps will be carrying out in this approach. The more the steps included, the more the accurate the requirements collected. 4 points for Time Effective Approach since 80% of the steps are included. As around 50% of the steps area included in EDI Application Approach, 3 points is assigned to this aspect.

The requirement collection time required to implement the recommended steps in Full Approach is relatively longer than the other two approaches. And with only limited resources available in our SDU, it’s not possible for my company to implement this Full Approach. Therefore only 1 point is given to Full Approach. As for the Time Effective Approach consumes less time for requirement collection, it favors to my company’s situation and hence 5 points are given. While EDI Application Approach lies in the middle, and 3 points is given to this approach.

SDU in Com-S has insufficient human resources, therefore it’s hard to have someone who can put a lot of time in requirement collection. In addition to time limitation, it’s also hard to process the Full Approach in Com-S’s application development, and hence only 2 points are given to Full Approach in Human Resources Aspect. For Time Effective Approach, fewer steps are involved when compared to the Full Approach and the need for human resources is relatively lower. Although it’s not the most optimal one, it’s much more applicable than the Full Approach, and hence 3 points are given. For the EDI Application Approach, since there are least steps are to be involved in the process and there are less interactions in-between users and stakeholders, less human resources are required in collecting requirements, as a result 5 points are given to EDI Application Approach.
When considering both the Time Effectiveness and Human Resources Effectiveness collectively, the cost of Full Approach is definitely the highest among the 3 approaches, and hence 2 points are given to Full Approach. Time Effective Approach and EDI Application Approach are found to be strong in Time Effectiveness and Human Resources Effectiveness respectively yet they are relatively weak on the other aspect, therefore both of them got 4 points.

For the Case Suitability Aspect, Full Approach seems not suitable for both cases I’ve mentioned above (Miscommunication with end-users & Miscommunication with 3rd parties). On the other hand, Time Effective Approach fits into the case of miscommunication with end-users since “Time is Money”, the least time used for collecting adequate requirement, the lower the cost and can generate more value afterwards. For the second case where miscommunication with 3rd parties is found, although problem users, as well as stakeholders are all well-defined at the very beginning, constraints that the system will be facing as well as non-functional requirements are still unknowns. Therefore, only EDI Application Approach is capable in solving the problem in case #2. As a result, 3, 4 & 4 points are given to Full Approach, Time Effective Approach and EDI Approach respectively.

Lastly is the Readiness For Implement Aspect. According to current situation, related forms, lists and implementation tools are not available at this moment. Furthermore, business use-case model and business object model are new to current SDU colleagues. Therefore all three approaches are not ready for implementation in a short period. 1 point is given to all 3 approaches.

Below is the table of evaluation which gives a more clear evaluation result of the three approaches,
follows is a spider-web chart which gives a graphical presentation on the evaluation result.

Evaluations on the three approaches

<table>
<thead>
<tr>
<th>Evaluation Aspect</th>
<th>Recommendation</th>
<th>Full Approach</th>
<th>Time Effective Approach</th>
<th>EDI Application Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step Completeness</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Time Effectiveness</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Human Resources Effectiveness</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Cost Effectiveness</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Case Suitability</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Ready For Implement</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td><em>Total</em></td>
<td>14</td>
<td>21</td>
<td>20</td>
<td></td>
</tr>
</tbody>
</table>
Implementation of Use-Case model to solve the miscommunication problem in requirement collection process

From the spider-web chart, both Time Effective Approach and EDI Application Approach are shown to be the solutions which can provide more values to my company. In addition to the fact that these two approaches are more suitable to solve the case problems, they also require less human resources and the costs involved are lower (i.e. more favor to my company and get a higher point). In conclusion, both Time Effective Approach and EDI Application approach are recommended to solve the miscommunication problems for case #1 and case #2 respectively.
Implementation

Pre-implementation actions

Before any implementation, related forms, lists, and tools are required to develop since these equipments are not readily available. Pre-implementation actions are listed below.

<table>
<thead>
<tr>
<th>Action</th>
<th>Related Parties</th>
<th>Days for development</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design / create “Problem Statement Form”</td>
<td>System Analyst</td>
<td></td>
</tr>
<tr>
<td>Design / create “Cause Prioritization List”</td>
<td>System Analyst</td>
<td></td>
</tr>
<tr>
<td>Design / create “User &amp; Stakeholder List”</td>
<td>System Analyst</td>
<td>5</td>
</tr>
<tr>
<td>Design / create “Constraint List”</td>
<td>System Analyst</td>
<td></td>
</tr>
<tr>
<td>Design / create sample context-free question set</td>
<td>Management Team</td>
<td></td>
</tr>
<tr>
<td>Design / create “Requirement Prioritization Form”</td>
<td>System Analyst</td>
<td>5</td>
</tr>
<tr>
<td>Design / create “Interview Guidelines”</td>
<td>System Analyst</td>
<td>4</td>
</tr>
<tr>
<td>Design / create supplementary documentation guidelines</td>
<td>Management Team</td>
<td></td>
</tr>
<tr>
<td>Design / create Fishbone diagram – tutorials</td>
<td>System Analyst</td>
<td>5</td>
</tr>
<tr>
<td>Design / create Business use-case model / Business Object model – tutorials</td>
<td>System Analyst</td>
<td>5</td>
</tr>
</tbody>
</table>
Furthermore, each new colleague is required to understand those forms and lists, read the guidelines and attend the tutorial classes. Below is the implementation plan for those new colleagues.

<table>
<thead>
<tr>
<th>Actions</th>
<th>Days for studies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Study all the forms, lists and guidelines</td>
<td>1</td>
</tr>
<tr>
<td>Fishbone diagram tutorial</td>
<td>1</td>
</tr>
<tr>
<td>Business use-case model &amp; Business Object model tutorial</td>
<td>2</td>
</tr>
</tbody>
</table>

**Implementation plan for case #1 - Time Effective Approach**

As mentioned in mini-case #1 in Mini-Case section, the miscommunication problem between SDU and intra-business unit end-users arose when there was request on new application development or update of existing applications. Since these were user-interactive applications and problems were not well defined, both Full Approach and Time Effective Approach are suitable for solving this miscommunication problem. In additional with the limited time constraint for development, Time Effective Approach is more preferable than Full Approach.

Time Effective Approach is recommended to use for solving the miscommunication problem with intra-business unit end-users, i.e. the problem discovered in case #1. The implementation of this approach is to follow the steps listed in recommendation section without any modifications. Below is the plan of implementation.
Steps required for case #1 implementation

<table>
<thead>
<tr>
<th>Stage / Steps</th>
<th>Related Parties</th>
<th>Tool(s)</th>
<th>Max. Days</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Stage 1: Analyzing the problem</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Gain Agreement on the problem with</td>
<td>System Analyst</td>
<td>Problem Statement Form</td>
<td>1</td>
</tr>
<tr>
<td>users and stakeholders</td>
<td>End-Users</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Understand the root causes</td>
<td>System Analyst</td>
<td>Fishbone Diagram</td>
<td>2</td>
</tr>
<tr>
<td>- Define the solution system boundary</td>
<td>System Analyst</td>
<td>Business Use-Case Model &amp; Business Object Model</td>
<td></td>
</tr>
<tr>
<td><strong>Stage 2: Understanding the users and stakeholders need</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Read Interview Guidelines</td>
<td>System Analyst</td>
<td>Interview Guideline N/A</td>
<td></td>
</tr>
<tr>
<td>- Developing Context-Free Questions</td>
<td>System Analyst</td>
<td>Context-Free Questions Set</td>
<td>3</td>
</tr>
<tr>
<td>- Interview with users &amp; stakeholders</td>
<td>System Analyst</td>
<td>Requirement Prioritization Form</td>
<td>1</td>
</tr>
<tr>
<td>End-Users</td>
<td>End-Users</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Process “Critical, Important &amp; Useful”</td>
<td>End-Users</td>
<td></td>
<td></td>
</tr>
<tr>
<td>categorization</td>
<td>End-Users</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Stage 3: Developing supplementary documentation</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Developing supplementary documentation</td>
<td>System Analyst</td>
<td>N/A</td>
<td>1</td>
</tr>
</tbody>
</table>

Total Working Days Required: 8
Implementation plan for case #2 - EDI Application Approach

The miscommunication problem mentioned in case #2 arose when there was a need for data interchange with Com-S’s partners. The problem for the applications and root causes were well defined in the beginning, but both the system boundary and system constraints are yet to be defined. When comparing the three recommendations, only EDI Application Approach fits in the situation where no problem and root cause definition is necessary, therefore EDI Application Approach is the only solution to this problem.

EDI Application Approach is recommended to use for solving the miscommunication problem between SDU and 3rd party stakeholders. Since 3rd party stakeholders may not be always available for interview, I will have to assign more working days for developing context-free questions in order to list out as many questions as possible and cover most of the area. So that we can get most of the answers ideally from a single interview with 3rd party stakeholders.

Steps required for case #2 implementation

<table>
<thead>
<tr>
<th>Stage / Steps</th>
<th>Related Parties</th>
<th>Tool(s)</th>
<th>Max. Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 1: Analyzing the problem</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Define the solution system boundary</td>
<td>System Analyst</td>
<td>Business Use-Case Model &amp; Business Object Model</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3rd Party Stakeholders</td>
<td>Constraint List</td>
<td></td>
</tr>
<tr>
<td>- Identify the constraints to be</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Steps required for case #2 implementation (Cont’d)

<table>
<thead>
<tr>
<th>Stage / Steps</th>
<th>Related Parties</th>
<th>Tool(s)</th>
<th>Max. Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 2: Understanding the users and stakeholders need</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Read Interview Guidelines</td>
<td>System Analyst</td>
<td>Interview Guideline</td>
<td>N/A</td>
</tr>
<tr>
<td>- Developing Context-Free Questions</td>
<td>System Analyst</td>
<td>Context-Free Questions Set</td>
<td>5</td>
</tr>
<tr>
<td>- Interview with users &amp; stakeholders</td>
<td>3rd Party Stakeholders</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Stage 3: Developing supplementary documentation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Developing supplementary documentation</td>
<td>System Analyst</td>
<td>N/A</td>
<td>2</td>
</tr>
<tr>
<td>Total Working Days Required:</td>
<td></td>
<td></td>
<td>12</td>
</tr>
</tbody>
</table>

Evaluation of the implementation plans

The implementation costs required for both Time Effective Approach and EDI Application Approach are relatively higher as more resources are needed for the requirement collection process. Before implementing both Time Effective Approach and EDI Application Approach to Case #1 and #2, the total number of working days needed for requirement collection is 3 and 4 days respectively. After implementation, the total number of working days needed is 8 and 12 days respectively.

Although the cost for requirement collection is higher, the quality of requirements collected is improved.
Before implementation, problem is not well defined, requirements collected were not well categorized, 
and therefore it was hard to discover any missing or un-identified requirements. As a result, we need to 
re-collect the missing requirements and to do modifications on the finished application.

After implementation, problem and its causes are well defined by the use of fishbone diagram and 
use-case models. Requirements are categorized into functional and non-functional aspects which help in 
identifying the missing ones. Interview guidelines help in ensure the quality of requirements collected too.

Combining all these improvements, there is no need to do system / application re-development, service 
level to customers is not affected and the cost of re-development is 0.

Below is the table of evaluation summary on the implementation of both approaches to case #1 & #2.

**Implementation Plan Evaluation Summary**

<table>
<thead>
<tr>
<th>Evaluation Aspect</th>
<th>Case #1</th>
<th>Case #2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Before</td>
<td>After</td>
</tr>
</tbody>
</table>
| Cost of collection      | 3 working days  
                         | :: low (in terms of resources consumed) | 8 working days  
                         | :: high (in terms of resources consumed) | 4 working days  
                         | :: low (in terms of resources consumed) | 12 working days  
                         | :: high (in terms of resources consumed) |
| Requirement Categorization | ✓       | ✓       | ✓       | ✓       |
| Quality of information / No standard to higher with | No standard to higher with | No standard to higher with |
Implementation of Use-Case model to solve the miscommunication problem in requirement collection process

<table>
<thead>
<tr>
<th>requirements collected</th>
<th>ensure quality</th>
<th>assurance</th>
<th>ensure quality</th>
<th>assurance</th>
</tr>
</thead>
</table>

Implementation Plan Evaluation Summary (Cont’d)

<table>
<thead>
<tr>
<th>Evaluation Aspect</th>
<th>Case #1</th>
<th>Case #2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Before</td>
<td>After</td>
</tr>
<tr>
<td>Missing requirement</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Requirement recollection</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Application re-development</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Cost of re-development</td>
<td>High (in terms of resources consumed &amp; decrease in quality of service)</td>
<td>0</td>
</tr>
</tbody>
</table>
Conclusion

Miscommunication is a major problem found in Com-S, especially when there is a need in developing new applications or updating existing applications. Insufficient or irrelevant information are always being gathered under the requirement collection process, as a result, the final outputs can only partially fulfill users and stakeholders’ requirements, i.e. the case of miscommunication with the end-users. For those electronic data interchange applications, there are always some unexpected non-functional incidents which affect Com-S’s level of services, i.e. the case of miscommunication with 3rd parties.

In order to overcome this problem, I can make use of some skills, framework or model that is available for improving the requirement collection process. I’ve reviewed the Use Case model and found that this is useful for solving the miscommunication problem of Com-S, therefore I’ll try to adopt this technique and apply it for improving the situation.

The Use Case Model (Dean & Don, 2003) consists of three stages for requirement collection, different skills and techniques are used in different stage. The first stage of the model is named “Analyzing the problem”. The focus of this stage is to identify and elaborate the problems. In order to discover the problems and the main causes, “Problem Statement Form” is used for listing the problems out, “Fishbone diagram” is used for describe the root causes. To define the boundary of the system, “Business use-case model” and “Business Object Model” are used for setting the system boundary in a graphical presentation. Moreover, a “Constraint List” is used for pointing out the constraints that the new system will have.

In the second stage, the model is going to “Understanding the users and stakeholders need”. Interview is
the main course in the stage, in order to make the interviews become more successful and collect adequate and useful information and requirements, “Interview Guidelines” is create for the interviewers to follow. Furthermore, “Sample Context-Free Questions”, which helps to minimize the bias from the interviewers, are needed to be prepared before any interviews. Collected requirements will be summarized after interviews and a prioritization process “Critical, Important & Useful Categorization” will be carry out to prioritize the importance of the requirements. Up to this stage, around 80% of all the requirements, which are functional requirements, are collected.

Lastly in the third stage, the model is going to collect the remaining 20% requirements, which are called non-functional requirements. The name of this stage is “Developing supplementary documentation”, this supplementary document records the data according to the categories usability, reliability, performance and supportability of the new system. Furthermore the design requirement and the non-functional constraints will be recorded in this documentation too.

With the help of the Use Case Model, a clear instruction set is created for every application development and system update. As adequate and useful information and requirements are being collected in the requirement collection stage, it will greatly improve the quality of the system being developed and minimize the opportunity of re-development of the existing applications, thus less time wasted in re-development process, resources can be used in other aspects. As a result, communication between end-users and stakeholders is improved, cost of application implementation is lowered and service level of Com-S is improved as well. An overall benefit is created and strengthened the compatibility of Com-S.
Self-Evaluation

The section of self-evaluation is use for evaluating my individual performance and is a continual assessment of me on the whole project. Logs for the time schedule, difficulties that I’ve faced throughout the project and how to overcome them will be shown in the following paragraphs.

Time Schedule

Below is the summary of the project team meetings. The learning team members will show up in each meeting, share their findings on their related projects as well as share their points of view on others’ findings and progresses.

Time Schedule and event highlights

<table>
<thead>
<tr>
<th>Date of meeting</th>
<th>Brief Description</th>
<th>Highlights</th>
</tr>
</thead>
<tbody>
<tr>
<td>19-Oct., 2008</td>
<td>1st Meeting</td>
<td>Introduction of project titles and brief explanations by team members. Ask for comments on the title from supervisor.</td>
</tr>
<tr>
<td>10-Nov., 2008</td>
<td>2nd Meeting</td>
<td>Report the findings and possible solutions to the problem. Share the case studies of each project. Update the writing progress of the report.</td>
</tr>
<tr>
<td>22-Jan., 2009</td>
<td>3rd Meeting</td>
<td>Progress reporting by team members. Share of the literature reviews.</td>
</tr>
</tbody>
</table>
Time Schedule and event highlights (Cont’d)

<table>
<thead>
<tr>
<th>Date of meeting</th>
<th>Brief Description</th>
<th>Highlights</th>
</tr>
</thead>
<tbody>
<tr>
<td>12-Mar., 2009</td>
<td>4th Meeting</td>
<td>Discuss the constraints discovered within learning group. Correct the points that are not in the right track.</td>
</tr>
<tr>
<td>7-Apr., 2009</td>
<td>5th Meeting</td>
<td>Update report writing progress. Discuss the recommendations of each teammate in their reports and share the points of views.</td>
</tr>
</tbody>
</table>

Personal Learning

The whole project starts with problem identification, follows by research for possible solutions, literature studies and review, recommendations and finally ends with implementation. Throughout the process, there are difficulties I need to overcome. The learning process on how to overcome the difficulties helps me to develop my managerial skills and enrich my academic knowledge.

Academically, I’ve learnt how Use Case Model helps in solving the miscommunication problem. Inside Use Case Model, I’ve understand the usage of Business Use Case Model and Business Object Model. Furthermore, Fishbone Diagram is a new technique I’ve learnt throughout this progress which is helpful in identifying the causes of the problems.

In managerial aspect, I’ve understand that apart from technical skills, guidelines for the procedures are needed in order to standardize and ensure the quality of outputs from each step. Predefined forms and lists can help in smoothing the information collecting process, and also help in providing highly readable and
understandable information no matter who is going to deliver this information.

Difficulties Faced and Overcome Solution

The very first difficulties I encountered is the lack of understanding about Use Case Model, therefore what I have to do in the beginning is to read some articles and get more knowledge of Use Case Model from books.

Since Use Case Model is talking about the whole process of software development, i.e. start from problem identification, requirements collection and selection, building system and system testing, it’s not relevant to implement the whole Model into my cases. I have to select those relevant components and focus on how to apply them to solve the miscommunication problem.

When applying those Use-Case components, I’ve found that different people will have different standards on the definition of “adequate and useful information”, which affects the readability and understandability of this information. It seems on top of applying these components, some kind of standards need to be implemented too. Therefore I’ve tried to create some forms and lists for standardizing the outputs from each step to overcome the readability and understandability of the information collected.

Furthermore, the quality of interviews is another issue that will greatly affect the performance of the requirements collection. In order to ensure the quality, and from the management point of view, a set of formal and instructive guidelines is needed.
Reference


