Universiteit Leiden

ICT in Business

Capability Maturity Model for Software Usage

Name: Yunwei Huang
Student-no: s1101005

Date: 16/06/2014

1st supervisor: Dr. Luuk Groenewegen

2nd supervisor: Dr. Nelleke Kooimans

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)

Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherland

Abstract

Users or customers sometimes are not quite satisfied with software product after its delivery. Some of them claimed that the software product was not like the way they described. This may because of the miscommunications or misunderstandings in requirements stage, or users/potential users may not have a clear idea what kind of software products they really want. So how to resolve this problem is the topic of this thesis. By building capability maturity model for software usage (CMM for SU), organizations could find out users' real requirements by learning users' behavior towards using software products. This model defined five maturity levels based on CMMI for SE, to explain what organizations should do in order to retrieve usage data, and how could they translate usage data into users' real requirements, and to improve software products based on users' real requirements.

Chapter 1: Introduction

With the development of IT industry, there are more and more options for people to choose when they are looking for software solutions. The key to win in this market is to obtain and to retain a good reputation. And reputation is always about customers. How to develop software that fit customers' real and potential needs, and then promote customers' satisfaction is very often a problem for software vendors. And software vendors are trying different ways to try to meet customers' real needs. But it has not worked well. It is true that customers and software vendors very often sign a service level agreement (SLA) and specify requirements of software. And there are also some methods applied trying to avoid misunderstanding within the organization (software vendor) itself, for example, using UML. But results turn out failed sometimes; this may because customers sometimes do not exactly know what they real requirements are. So how to dig the real needs of customers is the key problem has to be resolved. There is one field trying to resolve this problem, which is CMMI for SE, where there exist some more. As you may know, CMMI for SE is a process improvement training and certification program and service administered and marketed by Carnegie Mellon University and required by many DOD and Government programs for government contracts, especially software development. [1] But CMMI is mainly focused on process improvement within the organization itself, so there are some missing parts in SE that could help to resolve the mismatch between customers and software vendors -- SU is one solution. By tracking the usage of software of customers, software vendors then could improve their products to be more aligned with the real needs of customers, and promote customers' satisfaction to obtain a good reputation. And only by continuously improving software vendors' products could retain customers and vendors' reputation, and expand their market.

So our research question here is:

How to improve software usage based on users' real behavior towards using software products instead of just asking customers or potential customers what their needs are.

In order to achieve the target, what we should do? For the research, we have set the following sub-goals. First of all, we have to understand what software usage is and why learning software usage is helpful for software vendors. Answering this question is our main goal. After we understood the importance of software usage, then we have to figure out how we could get information about software usage. Here, we will introduce a model (CMMI) for software engineering, which will help us build a model to learn software usage. And based on CMMI for

SE, we will study Capability Maturity Model for Software Usage (CMM for SU) to help vendors learning users' behavior towards using software and making improvements based on the knowledge of software usage.

Based on investigating these topics and on achieving these sub-goals accordingly, this thesis has the following structure. There are 7 chapters included. We have chapter 1 for introduction, then followed by chapter2, in which we will discuss the importance of SU in software industry. And in chapter 3, we will briefly introduce CMMI. Then we will explain the maturity model of SU in chapter 4. There are five levels in the model: level 1—Ad Hoc; level 2—Basic Supporting; level 3—Standardization; level 4—Room for Improvement; and level 5 Optimizing. We will find out what the characteristics of each level are, what the differences between different levels are, and what components/processes should be included in order to achieve each maturity level. In chapter 5, we stand back and formulate an overview by comparing CMM for SU and CMMI for SE. In the last chapter we will have the conclusion of CMM for SU. Chapter 6 concludes by summarizing the model. And in chapter 7, we indicate some future research possibilities concerning this topic.

Chapter 2: Importance of SU

What is SU and how SU can help software vendors with their software products? As we mentioned earlier, SU stands for software usage, and tracking software usage could help us learn about software usage. For example, after users purchased software, vendors may want to know how customers feel about their software so that they could make changes to their software products, in order to improve software products quality and customer satisfaction, thus expanding their markets. But the problem here is that users normally refuse to give feedback about software products unless they have some problems occurred during the usage of software product. And even though users are agreed to give feedback about using software, those vendors may still not satisfied feedback because most of end users are not professional develop engineers, they could not give "professional" suggestions about software products. So how to retrieve effective feedback from end users? Tracking Software Usage is one of the possible solutions. Instead of passively retrieve information from end users, software vendors could collect information from users by tracking their real behavior with users' approval: e.g. we could track which functions users use the most, and which functions they barely use; or what's the sequences of functions they use to find out which software functions/components meet the real requirements of users or failed to meet users' requirements, instead of asking users to fill a form or questionnaire to get feedback from users. And the information we get from monitoring users behavior is much more stable because that is users' real behavior, which can avoid miscommunication and misunderstanding problems. As situations may be different within different organizations, we defined 5 different capability maturity levels for tracking software usage and make use of the usage data we collected from users by tracking software usage. Organizations could decide which level they want to achieve according to their current situation. Therefore, the information collected is rather unique as well as valuable.

Another important reason of learning software usage is that it promotes efficiency. Traditional ways of retrieving feedback from users may take some days to prepare interview questions or questionnaires, whereas tracking software usage could help software vendors to retrieve information they needed immediately, without waiting time. Therefore, it saves time to get requirements both for software vendors and users.

Chapter 3: Introduction of CMMI for SE

We will firstly have a brief introduction of CMMI for SE, and then followed by explaining each maturity level and its process areas in order to achieve each maturity level. Please keep in mind that this entire chapter is about capability maturity of the software engineering process and not of software usage.

3.1 What is CMMI?

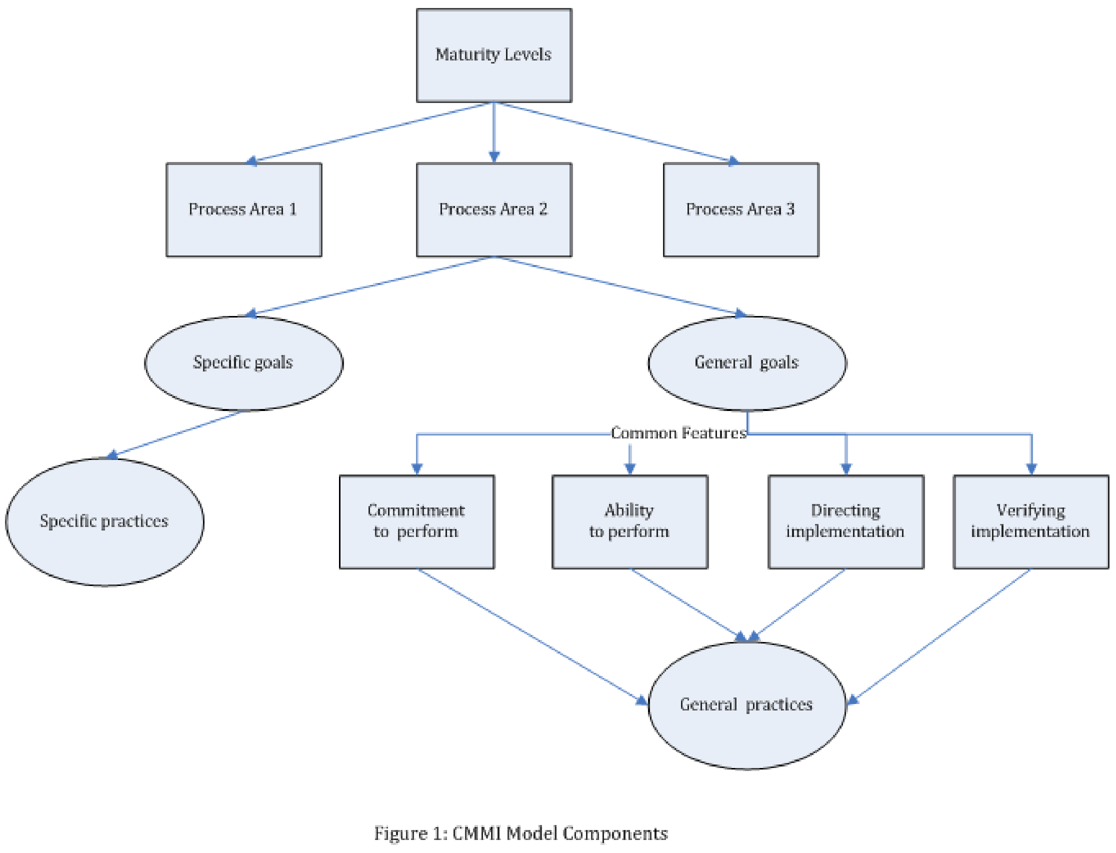


Figure 3-1: CMMI Model Components

The best definition of CMMI is from the document which was written and updated by Carnegie Mellon Software Engineering Institute. So I will briefly introduce CMMI models by citing the content from the document they published. For more information about CMMI for SE, you can visit their website for more information or you could review their document through the following link: http://www.sei.cmu.edu/reports/02tr012.pdf.

CMMI models are designed to describe discrete levels of process improvement. In the staged representation, maturity levels provide a recommended order for approaching process improvement in stages. As illustrated in Figure 1, maturity levels organize the process areas. Within the process areas are generic and specific goals as well as generic and specific practices. Common features organize generic practices [2].

This representation focuses on best practices your organization can use to improve processes in the process areas that are within the maturity level it chooses to achieve. Before you begin using a CMMI model for improving processes, you must map your processes to CMMI process areas. This mapping enables you to control process improvement in your organization by helping you track your organization's level of conformance to the CMMI model you are using. It is not intended that every CMMI process area maps one to one with your organization's processes [3].

3.2 What are the maturity levels?

Material in this subsection comes from [4].

The maturity level of an organization provides a way to predict the future performance of an organization within a given discipline or set of disciplines. Experience has shown that organizations do their best when they focus their process-improvement efforts on a manageable number of process areas that require increasingly sophisticated effort as the organization improves.

A maturity level is a defined evolutionary plateau of process improvement. Each maturity level stabilizes an important part of the organization's processes.

In CMMI models with a staged representation, there are five maturity levels, each a layer in the foundation for ongoing process improvement, designated by the numbers 1 through 5:

1. Initial
2. Managed
3. Defined
4. Quantitatively Managed
5. Optimizing

3.3 Details of each Maturity Level

Material in this subsection comes from [5].

3.3.1 Maturity Level 1: Initial

At maturity level 1, processes are usually ad hoc and chaotic. The organization usually does not provide a stable environment. Success in these organizations depends on the competence and heroics of the people in the organization and not on the use of proven processes. In spite of this ad hoc, chaotic environment, maturity level 1 organizations often produce products and services that work; however, they frequently exceed the budget and schedule of their projects.

Maturity level 1 organizations are characterized by a tendency to over commit, abandon processes in the time of crisis, and not be able to repeat their past successes.

3.3.2 Maturity Level 2: Managed

At maturity level 2, an organization has achieved all the specific and generic goals of the maturity level 2 process areas. In other words, the projects of the organization have ensured that requirements are managed and that processes are planned, performed, measured, and controlled.

The process discipline reflected by maturity level 2 helps to ensure that existing practices are retained during times of stress. When these practices are in place, projects are performed and managed according to their documented plans.

At maturity level 2, requirements, processes, work products, and services are managed. The status of the work products and the delivery of services are visible to management at defined points (for example, at major milestones and at the completion of major tasks).

Commitments are established among relevant stakeholders and are revised as needed. Work products are reviewed with stakeholders and are controlled. The work products and services satisfy their specified requirements, standards, and objectives.

3.3.3 Maturity Level 3: Defined

At maturity level 3, an organization has achieved all the specific and generic goals of the process areas assigned to maturity levels 2 and 3. At maturity level 3, processes are well characterized and understood, and are described in standards, procedures, tools, and methods.

The organization’s set of standard processes, which is the basis for maturity level 3, is established and improved over time. These standard processes are used to establish consistency across the organization. Projects establish their defined processes by tailoring the organization’s set of standard processes according to tailoring guidelines.

The organization’s management establishes process objectives based on the organization’s set of standard processes and ensures that these objectives are appropriately addressed.

A critical distinction between maturity level 2 and maturity level 3 is the scope of standards, process descriptions, and procedures. At maturity level 2, the standards, process descriptions, and procedures may be quite different in each specific instance of the process (for example, on a particular project). At maturity level 3, the standards, process descriptions, and procedures for a project are tailored from the organization's set of standard processes to suit a particular project or organizational unit. The organization's set of standard processes includes the processes addressed at maturity level 2 and maturity level 3. As a result, the processes that are performed across the organization are consistent except for the differences allowed by the tailoring guidelines.

Another critical distinction is that at maturity level 3, processes are typically described in more detail and more rigorously than at maturity level 2. At maturity level 3, processes are managed more proactively using an understanding of the interrelationships of the process activities and detailed measures of the process, its work products, and its services.

3.3.4 Maturity Level 4: Quantitatively managed

At maturity level 4, an organization has achieved all the specific goals of the process areas assigned to maturity levels 2, 3, and 4 and the generic goals assigned to maturity levels 2 and 3. Subprocesses are selected that significantly contribute to overall process performance. These selected subprocesses are controlled using statistical and other quantitative techniques.

Quantitative objectives for quality and process performance are established and used as criteria in managing processes. Quantitative objectives are based on the needs of the customer, end users, organization, and process implementers. Quality and process performance are understood in statistical terms and are managed throughout the life of the processes.

For these processes, detailed measures of process performance are collected and statistically analyzed. Special causes of process variation2 are identified and, where appropriate, the sources of special causes are corrected to prevent future occurrences.

Quality and process performance measures are incorporated into the organization's measurement repository to support fact-based decision making in the future.

A critical distinction between maturity level 3 and maturity level 4 is the predictability of process performance. At maturity level 4, the performance of processes is controlled using statistical and other quantitative techniques, and is quantitatively predictable. At maturity level 3, processes are only qualitatively predictable.

3.3.5 Maturity Level 5: Optimizing

At maturity level 5, an organization has achieved all the specific goals of the process areas assigned to maturity levels 2, 3, 4, and 5 and the generic goals assigned to maturity levels 2 and 3. Processes are continually improved based on a quantitative understanding of the common causes of variation3 inherent in processes.

Maturity level 5 focuses on continually improving process performance through both incremental and innovative technological improvements. Quantitative process-improvement objectives for the organization are established, continually revised to reflect changing business objectives, and used as criteria in managing process improvement. The effects of deployed process improvements are measured and evaluated against the quantitative process-improvement objectives. Both the defined processes and the organization's set of standard processes are targets of measurable improvement activities.

Process improvements to address common causes of process variation and measurably improve the organization's processes are identified, evaluated, and deployed. Improvements are selected based on a quantitative understanding of their expected contribution to achieving the organization's process-improvement objectives versus the cost and impact to the organization. The performance of the organization's processes is continually improved.

Optimizing processes that are agile and innovative depends on the participation of an empowered workforce aligned with the business values and objectives of the organization. The organization's ability to rapidly respond to changes and opportunities is enhanced by finding ways to accelerate and share learning. Improvement of the processes is inherently part of everybody's role, resulting in a cycle of continual improvement.

A critical distinction between maturity level 4 and maturity level 5 is the type of process variation addressed. At maturity level 4, processes are concerned with addressing special causes of process variation and providing statistical predictability of the results. Though processes may produce predictable results, the results may be insufficient to achieve the established objectives. At maturity level 5, processes are concerned with addressing common causes of process variation and changing the process (that is, shifting the mean of the process performance) to improve process performance (while maintaining statistical predictability) to achieve the established quantitative process improvement objectives.

3.4 Four Categories of CMMI Process Areas

Material in this subsection comes from [6].

Process areas can be grouped into four categories:

- Process Management
- Project Management

- Engineering
- Support

3.4.1 Process Management Process Areas

Process Management process areas contain the cross-project activities related to defining, planning, resourcing, deploying, implementing, monitoring, controlling, appraising, measuring, and improving processes

The Process Management process areas of CMMI are as follows:

- Organizational Process Focus
- Organizational Process Definition
- Organizational Training
- Organizational Process Performance
- Organizational Innovation and Deployment

3.4.2 Project Management Process Areas

Project Management process areas cover the project management activities related to planning, monitoring, and controlling the project.

The Project Management process areas of CMMI are as follows:

- Project Planning
- Project Monitoring and Control
- Supplier Agreement Management
- Integrated Project Management for IPPD (or Integrated Project Management)
- Risk Management
- Integrated Teaming
- Integrated Supplier Management
- Quantitative Project Management

3.4.3 Engineering Process Areas

Engineering process areas cover the development and maintenance activities that are shared across engineering disciplines (e.g., systems engineering and software engineering). The six process areas in the Engineering process area category have inherent interrelationships. These interrelationships stem from applying a product development process rather than discipline-specific processes such as software engineering or systems engineering.

The Engineering process areas of CMMI are as follows:

- Requirements Development
- Requirements Management
- Technical Solution
- Product Integration
- Verification
- Validation

3.4.4 Support Process Areas

Support process areas cover the activities that support product development and maintenance. The Support process areas address processes that are used in the context of performing other processes. In general the Support process areas address processes that are targeted towards the project, and may address processes that apply more generally to the organization. For example, Process and Product Quality Assurance can be used with all the process areas to provide an objective evaluation of the processes and work products described in all of the process areas.

The Support process areas of CMMI are as follows:

- Configuration Management
- Process and Product Quality Assurance
- Measurement and Analysis
- Organizational Environment for Integration
- Decision Analysis and Resolution
- Causal Analysis and Resolution

3.5 Process Areas for each Maturity Level

Material in this subsection comes from [7].

3.5.1 Process Areas for Maturity Level 2

The following section contains all of the process areas that belong to maturity level 2. The maturity level 2 process areas of CMMI are as follows:

- Requirements Management

- Project Planning
- Project Monitoring and Control
- Supplier Agreement Management
- Measurement and Analysis
- Process and Product Quality Assurance
- Configuration Management

3.5.2 Process Areas for Maturity Level 3

The following section contains all of the process areas that belong to maturity level 3. The maturity level 3 process areas of CMMI are as follows:

- Requirements Development
- Technical Solution
- Product Integration
- Verification
- Validation
- Organizational Process Focus
- Organizational Process Definition
- Organizational Training
- Integrated Project Management for IPPD
- Risk Management
- Integrated Teaming
- Integrated Supplier Management
- Decision Analysis and Resolution
- Organizational Environment for Integration

3.5.3 Process Areas for Maturity Level 4

The following section contains all of the process areas that belong to maturity level 4. The maturity level 4 process areas of CMMI are as follows:

- Organizational Process Performance
- Quantitative Project Management

3.5.4 Process Areas for Maturity Level 5

The following section contains all of the process areas that belong to maturity level 5. The maturity level 5 process areas of CMMI are as follows:

- Organizational Innovation and Deployment
- Causal Analysis and Resolution

CMMI for SE is the base of CMM for SU. They are similar in essence, but also have many differences. There are similar process areas between maturity model for software engineering and for software usage, but we could not map every process areas from CMMI for SE to CMM to SU. This is firstly because those two models focus on different aspects. CMMI for SE focuses on software engineering whereas CMM for SU focuses on software usage. For example, some software engineering process areas mentioned in CMMI for SE may not relevant for CMM for SU, as CMM for SU mainly focus on software usage, and software engineering is not our biggest concern. Secondly, since there are limited resources could be used for this thesis, we could not cover all possible process areas for each maturity level, thus, for each maturity level for software usage, we will only discuss the process areas which matter the most to us. Thirdly, there will be some new process areas which are not included in CMMI for SE, as those process areas are important regarding to software usage, but not important for software engineering.

In addition, we will be using "Archi", which is an ArchiMate modeling tool, to demonstrate the life cycle of each maturity level for software usage. The main reason we use "Archi" to demonstrate life cycle of each maturity level for software usage is that, unlike software engineering, there are no specific software to demonstrate the processes of tracking software usage, and we believe that representing business architecture and IT architecture of an organization could help us to explain how tracking software usage works. And "Archi" is a free software which could meet our needs. You could find the basic information about "Archi" in appendix 2 and if you are interested in this tool and want to find more detailed information about "Archi", you could visit their website: http://archi.cetis.ac.uk/.

Chapter 4: Introduction of CMM for SU

As we mentioned earlier, CMMI for SE is a process improvement training and certification program and service administered and marketed by Carnegie Mellon University and required by many DOD and Government programs for government contracts, especially software development. [1] CMMI for SE has defined five levels, which is the base of the Capability Maturity Model for SU.

4.1 Maturity Level 1

4.1.1 Origin of Maturity level 1

Maturity level 1 of CMMI is: Initial

At maturity level 1, processes are usually ad hoc and chaotic. The organization usually does not provide a stable environment. Success in these organizations depends on the competence and heroics of the people in the organization and not on the use of proven processes. Maturity level 1 organizations often produce products that work but exceed budget and schedule of projects. They are not being able to repeat their past success as there is no systematical way to guide them how to manage a project. [3]

4.1.2 Maturity level 1: Ad Hoc

Organizations at this level have no sense of what added value could be achieved by tracking what customers use their products for and how they use it, or they only have general customers who will not contact the organization again after purchase their products unless problems occurred when using their products. In another word, those organizations do not have or they do not think they have long-term relationship with their customers. As a matter of this, they only have support department to passively collect information from users, without understanding how users use their products.

4.1.3 Characteristics of level 1

- No/Weak relationship with customers
- Technical supports are provided to customers
- Passively receive feedback/advices/suggestions from customers about their products
- Improvement is not necessarily based on customers’ feedback

4.1.4 Life cycle of level 1

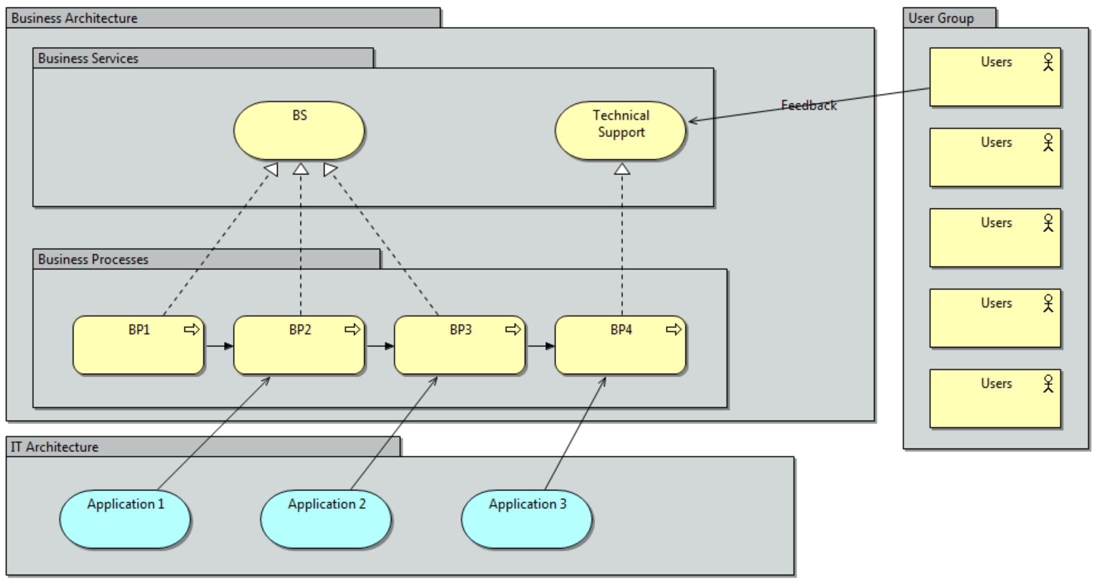


Figure 4-1-1: General Life Cycle of Maturity Level 1

From the diagram above, we could see how organizations at level 1 look like. We only list 2 simple business services in the diagram above (BS and Technical Support), but there could be many business services in an organization in real life situation. Suppose BS 2 includes many processes (BP1, BP2, BP3 and BP4), with or without the support of IT systems. For this particular diagram, application 1 and 2 are supporting BP2 and BP3 respectively, and BP1 doesn't need the support of IT system. After go through those processes, there is a support process (BP4) which is responsible to collect feedback from customers, which is supported by Application 3. From the diagram above, we could see that the technical support service is the only way the organization to connect with users. Therefore, organizations at this level have loose relationship with their customers. They only passively get information from customers. Feedback from customers has little/no influence for software engineering.

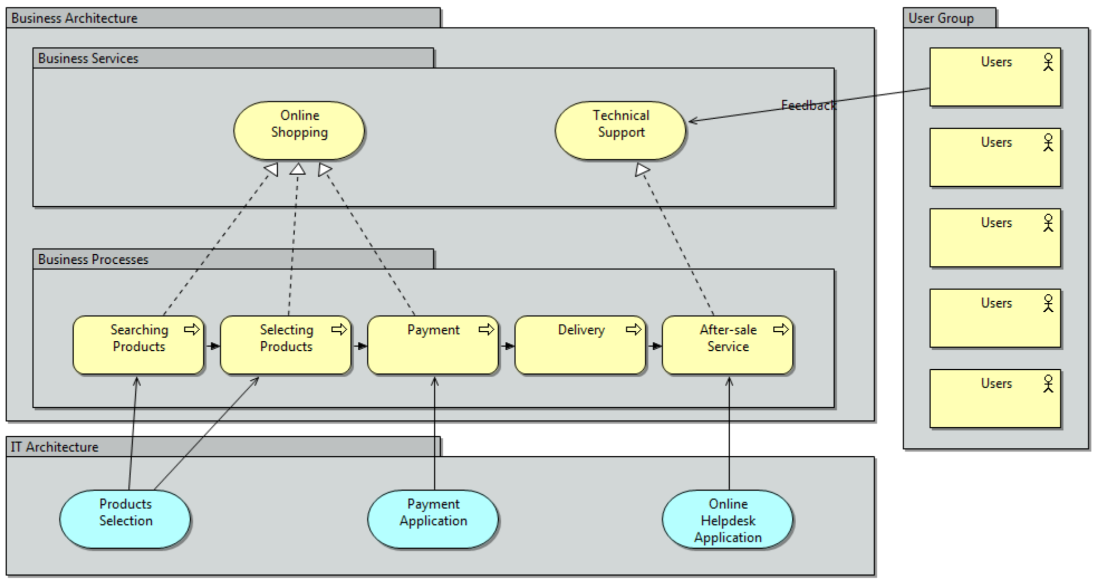


Figure 4-1-2: Example of Online Shopping Life Cycle of Maturity Level 1

Let’s take an online shopping website for example. There are at two business services which are supported by five business processes, namely searching products, selecting products, payment, delivery and after-sale service. The searching and selection processes are supported by products selection application, payment process is supported by payment application, and after-sale service process is supported by online helpdesk application. There is no application support delivery process. Customers will browse/search the products they are interested in, and then pay for them. There are two methods that customer will give feedback about the website. Either occurring problems during customers’ usage of the website, or they are willing to fill the survey to help improve the website after they finished their purchase of products. Either way, organizations at this level only passively receive information from customers.

4.2 Maturity Level 2

4.2.1 Origin of Maturity level 2

As we mentioned in 3.5.1, the process areas of CMMI maturity level 2 are as follows:

- Requirements Management
- Project Planning
- Project Monitoring and Control
- Supplier Agreement Management
- Measurement and Analysis
- Process and Product Quality Assurance
- Configuration Management

For CMM for SU level 2, the main focus is to define the process of collecting usage data. There are many processes involved, which we will discuss later. But before we discuss the processes, we first have to understand what organizations at level 2 want to achieve.

Organizations at this level have the initiation of tracking software usage. Therefore, the task is not so complicated. Their target is to design a tracking tool to monitor users' actions, and provide basic support whenever users encounter unexpected problems. And the tracking tool restores the usage data for future analysis and prevention of those problems happens again in the future.

4.2.1.1 Usage Requirements and their Management

Requirements managements should be part of the process areas of CMM for SU maturity level 2. In order to get information about software usage, we have to find a solution to collect data about what users use the products for, and how they use the products. There are many possible ways to collect information, and using tracking tool is an efficient and direct way to do this. Therefore, organizations at level 2 have to design their tracking tools and integrate into their software products, which mean they have to find out the requirements for those tracking tools. Here are the requirements they have to define:

1. Usage aspects requirement.

   For this requirement, organizations have to figure out which usage aspects they want to collect information from users

2. Functional requirements for tracking tools.

For this requirement, in order to fulfill the usage aspects requirements, organizations have to decide what the functional needs are for each tracking tool

3. Collecting data period requirement.

   For this requirement, organizations have to decide what the frequencies are to collect usage data: per week, per month or per day for each project

4. Sending data period requirements.

   For this requirement, organizations have to decide what the frequencies are to send back usage data

Another important part of this process area is to manage their usage requirements. The following are the aspects they should be included in order to manage their usage requirements:

1. Version management

In order to avoid the confusion of different usage requirements, organizations should establish rules to name the usage requirements. So that we could easily find out the usage requirements we needed by the name of documents.

2. Usage requirements dependency management

Updates of usage requirements should specify from which version of previous/historical usage requirements they made update. And they also have to specify which updates they want to make for the new version of usage requirements.

3. Storage management

Organizations have to define where to store the usage requirements. There could be many possible ways. Organizations could either mange the usage requirements per project, or they could create a separate storage room for managing all those usage requirements.

4.2.1.2 Usage Planning

For CMMI, project planning is a process area at maturity level 2. But for CMM for SU, instead of focusing on project, we mainly focus on software usage. Therefore, the process area should be usage planning.

For usage planning process area, we have to define what kind of usage we want to learn from users, and how we could collect the usage information. Here are the things you should include for your usage plan:

1. Usage requirements

Those requirements are what we mentioned in chapter 4.2.1.1

2. Project plan for developing tracking tool

For this part, you have to develop a project plan. For more information about how to develop a project plan, you can visit page 96 of the article: http://www.sei.cmu.edu/reports/02tr012.pdf

3. Usage data collection and data storage

For this part, you have to define how to collect usage data and where to restore the data

4. Usage data analysis

For this part, you have to define what your expected results are and what the real usage data are, and by analyzing the expected and real data, you could find a solution to satisfy your usage requirements.

In another word, those questions should be answered in your usage plan:

1. What usage aspects you want to learn and understand?

2. Why do you want to learn those usage aspects?

3. How do you translate those usage aspects into tracking functions?

4. What are the expected results of those usage aspects?

5. What are the possible solutions if the usage data you collect from users are quite different from your expected results?

4.2.1.3 Usage Monitoring

Again, instead of focusing on project monitoring, we mainly focus on usage. So usage monitoring is the area we concerned about.

As we mentioned earlier, there are several possible ways to collect information from users. You can do market research, surveys, etc. I would not say tracking tool is the best to do this; but it is an efficient way to finish the task. First of all, as we mentioned in chapter 2, instead of passively receive information from users, which I mean by doing surveys or questionnaires, you could proactively collect information from users; therefore, the data you collect is much reliable as you could avoid miscommunications and ambiguity. Secondly, instead of waiting response from users, you could collect the data whenever and wherever you want, which gives you much more flexibility than traditional method to retrieve data from users.

For this area, you have to understand what users' behavior you want to monitor, obviously you cannot monitor every action they made by using your software products. Technically, it is possible, but the data would be enormous and the costs would be too much to develop a tracking tool to track every action and restore the data. Therefore, the most important thing of usage monitoring is that you have to define what users' behavior you want to learn and monitor. You can get the information from your Usage Planning, which should already include the details of usage aspects that you want to track and learn.

After you know what you want to track, the next step you have to do is to develop your tracking tool according to your tracking needs, which we already explained the requirements of developing tracking tools in chapter 4.2.1.1.

4.2.1.4 Users Agreement Management

Unlike CMMI, we do not need a supplier; the most important factor here is users. We cannot get any information we need from users without their agreement. So it is important to sign an agreement with users according to your local laws.

For each your software product, if you want to track the software usage from users, you have to sign an agreement with users. Alternatively, you can just sign one agreement to collect data from any software products belong to your organization.

4.2.1.5 Measurement and Analysis

As we already mentioned in usage planning process area, you have to understand what you have to do with your tracking data. Otherwise, it is just a waste.

For tracking data, which is the usage data collected from users, there are many possible ways to measure it. The easiest way would be setting a usage standard for each usage aspect so that you could measure the data collected from users. But the problem is according to what you set the standard. If you have done marketing research before, you could use the result of marketing research as your standard. What should you do if you haven't done this before? An industry benchmark line could be an optical way to help set your usage standards. The third method is that you collect data information from users, and set your standards according to those data, and amend the standards after you collect more data from users. The last method has the least stability and it would take a long period to set the final standards, but your final standards would be the best fit to measure the data collected from your own users.

After the standards has been set, there should be someone or a team to analyze the usage data collected from users according to your usage standards. If the usage data has some deviations from your standards, you have to find out the possible reasons cause those deviations, and come up with solutions to help usage data achieve those standards. But there is another possibility, which is, your standards is somehow you cannot achieved by applying current technology, in this case, you have to alter your standards to a certain level that your usage data could be achieved. So if you tried many possible changes of your software products, and your usage data still could not achieve your usage standards, it would be a smart move to think if your usage standards are achievable or not. If not, try to revise them.

For CMMI, there are two more process areas are included at maturity level 2, namely process and products quality assurance, and configuration management. As for CMM for SU maturity level 2, the mainly focus is on software usage, in another word, we only focus on usage data instead of focusing on software engineering. Therefore, those two software engineering process areas are not included in the process areas of CMM for SU.

4.2.2 Maturity level 2: Basic Supporting

Organizations at this level know what value could be added by tracking what customers use their products for and how they use them. So it is important to design tracking tools to collect data they needed from customers, instead of passively receive feedback from them. Therefore, organizations at this level integrate tracking tools into their projects, and for every project, their own tracking tools are developed according to their specific requirements.

The most important characteristic of this level is that organization starts to develop their tracking tools to collect data, and integrate those tracking tools into their software products.

Compare to level 1, the advantages are obvious. As customers sometimes have wrong understanding of the products, or not quite sure how to describe problems in a way both customers and software vendors could understand, using tracking tools could help organizations eliminate the ambiguity and confusion to a large extent, and also help organizations understand how customers react with their products--what customers use their products for and how they use them. Therefore, organizations could improve their products based on the real needs of customers, not only from users' opinions.

4.2.3 Characteristics of level 2

- Usage requirements are required and documented for each software project
- Tracking tools are designed according to usage requirements and usage planning
- Tracking tools are integrated into their products to monitor and learn users' behavior
- Users Agreement for usage tracking should be included in each software

they want to track

- Measurement standards should be set
- Usage data collected from users should be analyzed by comparing with measurement standards

4.2.4 Life cycle of level 2

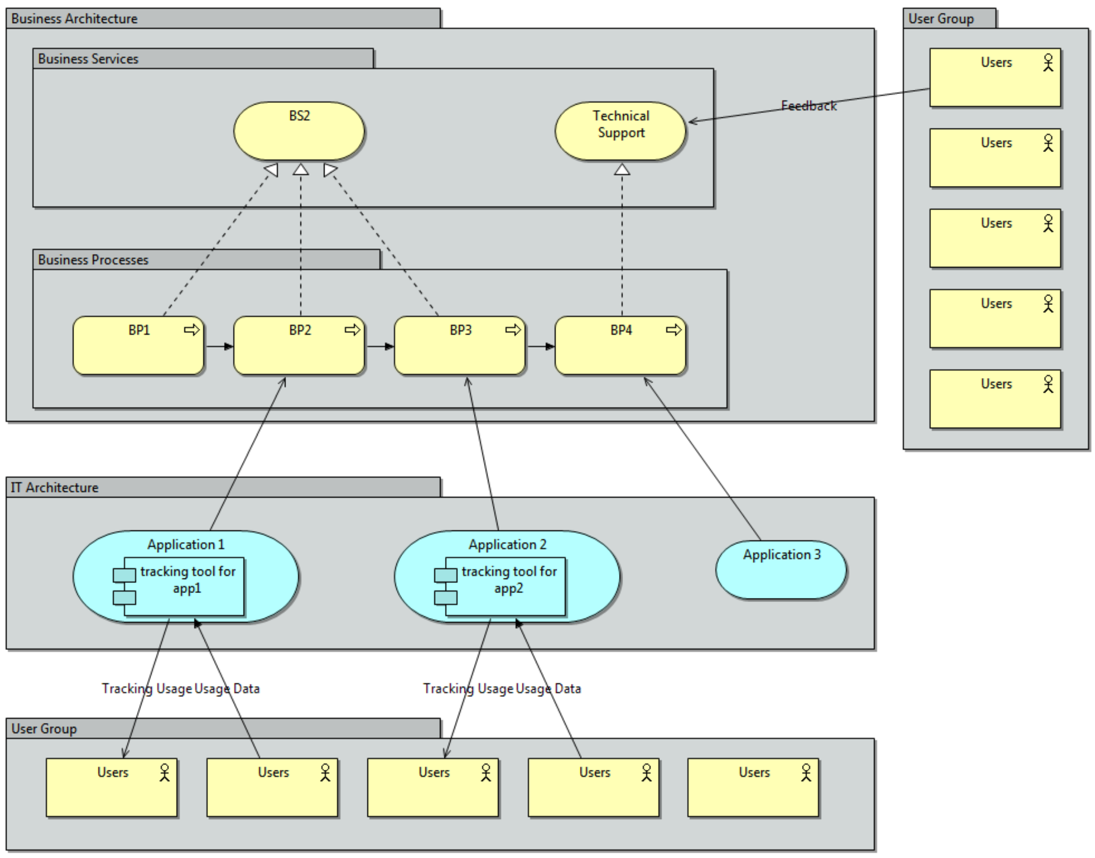


Figure 4-2-1: General Life Cycle of Maturity Level 2

The most obvious difference from level 1 is that organizations at this level start to track customers' usage by using tracking tools for single project instead of only receiving information from customers from their feedback or surveys.

As we can see from the diagram above, organizations at level 2 have differentiated from level 1. The differences are, first of all, tracking tools are integrated into their applications. Secondly, tracking tools are proactively tracking how users use their applications, and retrieving tracking data from users. By analyzing tracking data, organizations then could make improvement for either their software engineering process or software applications.

Let's take a simple online shopping for example:

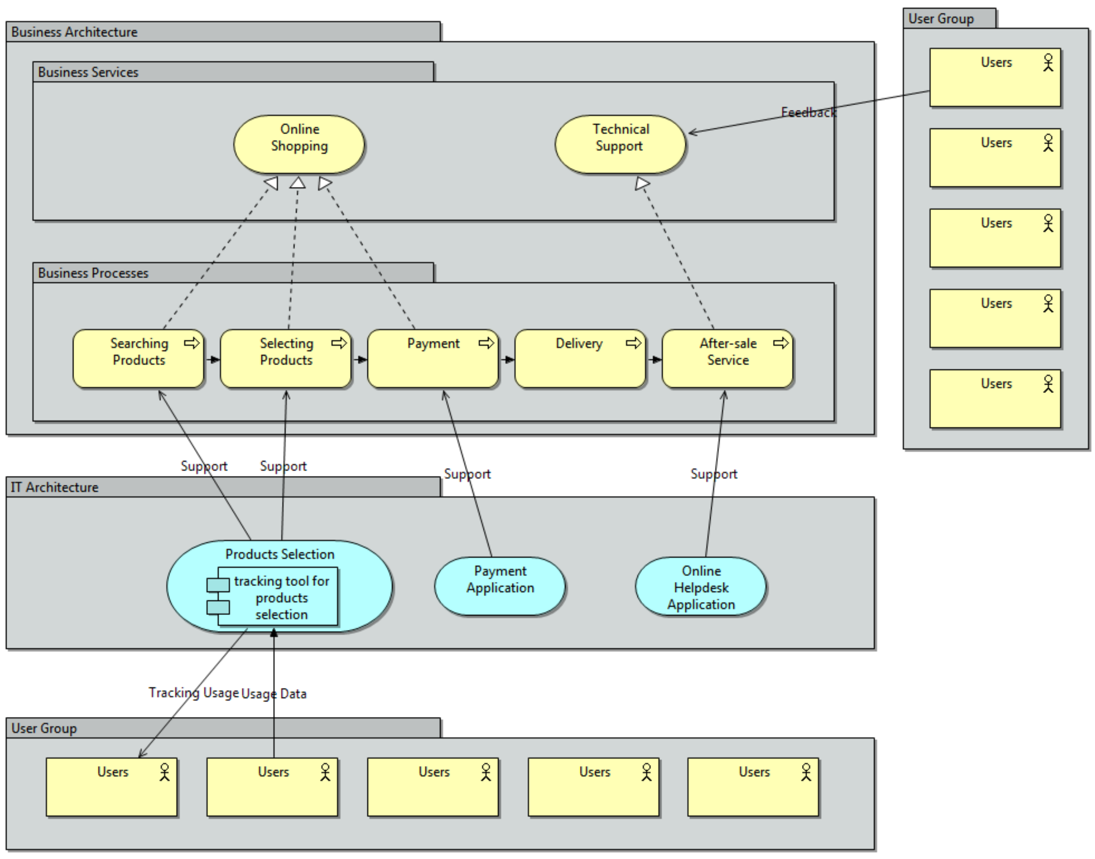


Figure 4-2-2: Example of Online Shopping Life Cycle of Maturity Level 2

As we mentioned above, the most important characteristic of maturity level 2 is tracking tool. In order to develop their tracking tool, organizations firstly have to develop their usage requirements according to chapter 4.2.1.1. Here are the things organizations have to do in order to develop their tracking tool.

- Define the usage aspects requirement

Suppose we want to find out what users do before they make the final decision to pay for the products

- Define functional requirements for tracking tool

We have to design a function to track and record users' behavior before they click to pay on the website

- Define collecting data period

We want to collect and restore data for each successful transaction

- Define sending data requirement

We want to receive the usage data every week

Those requirements should be documented. We should have one separate document for each project.

Once tracking tool has developed, they have to integrate it into product selection application. Then tracking tool could retrieve information about which functions of this selection application users use before they click to pay on the website. And based on those data, organization could do many decisions according to chapter 4.2.1.5. For example, they could decide to eliminate the functions with lowest using rate. Or they could make marketing strategy according to users' behavior. The point is, no matter what those usage data could be used for, the most important thing here is to collect usage data.

4.3 Maturity Level 3

4.3.1 Origin of Maturity level 3

The maturity level 3 process areas of CMMI are as follows:

- Requirements Development
- Technical Solution
- Product Integration
- Verification
- Validation
- Organizational Process Focus
- Organizational Process Definition of the SE process
- Organizational Training
- Integrated Project Management for IPPD
- Risk Management
- Integrated Teaming
- Integrated Supplier Management
- Decision Analysis and Resolution
- Organizational Environment for Integration

For organizations at CMM for SU maturity level 3, they have to standardize their methods to track software usage, and to check if users' behavior towards using software products is in the right way as the software products are designed. The focus here is to check the consistency between users' real behavior and their predicted behavior.

As software products are developed according to software requirements, what we do not know is that if those requirements are representing users' real needs or not. By monitoring users' behavior towards using software products, we could understand users' real needs. Then we could make adjustment of software functions to align with users' behavior, which presents their real needs.

The following are the processes have to be done in order to help organizations check the consistency of users' actual behavior and their predicted behavior, then find out users' real needs and make correction of software products in order that it more fit with users' real needs.

4.3.1.1 Usage Requirements Development

As we mentioned before, organizations at level 3 have to define a standard to develop software usage requirements, because only with this standard they could define a general tracking tool that could be applied to similar projects. In this way, they could collect more data from more users, so that they could set usage standards to evaluate if the usage has achieved the standards or not, if not, they could find some solution to achieve the standards. Those standards could not only be applied to the current projects in an organization, but also could evaluate their future projects.

In order to collect more usage data from users and set a valid usage standard for software usage, software usage requirements are crucial. We could learn from the definition of requirements development of CMMI, the purpose of requirements development is to produce and analyze customer, product, and product-component requirements. As for CMM for SU, the purpose of requirements development is slightly different, as its purpose is to produce and analyze usage requirements.

So what should organizations do to produce and analyze the usage requirements?

- Step1: Define what usage data they want to collect. This step is the most important part. Organizations have to define the requirements of usage data based on the requirements of software product itself. So that they could check if each function has achieved its goals or not.
- Step 2: Define how to collect those usage data we mentioned in step 1. As usage data may be collected from different components of software products, the way to collect data from different components might be different. Therefore, organizations at this level have to define functions which should be included in tracking tools in order to collect the usage data organizations needed.
- Step 3: Define where to where to store the usage data. There are many options; organizations could store all tracking data in one database, or they could choose to store their data in different database according to different projects. Alternatively, they could choose to store the data of similar projects in one database. It is their decision to make. Every option has its own advantages and disadvantages. The point here is organizations have to define in which way they choose to store their tracking data.
- Step 4: Define how to analyze the tracking data. First of all, for each function, they have to check if users' behavior is consistent with software requirement or not. If not, they have to find out the differences and what are the reasons causing those differences. If yes, then they also have to

define what the standard and acceptable range of usage rates for each function. Secondly, they have to define what the solutions for the functions that have not achieved the standard or not within the acceptable range of usage rates are.

By defining the usage requirements, organizations then could understand what the functional needs of tracking tools for each project are.

4.3.1.2 Technical Solution

For this part, it is the same as CMMI, except CMMI is project wise whereas CMM for SU is only focus on software usage. The process area of CMM for SU focuses on the following:

- Evaluating and selecting solutions for developing tracking tools

  As we mentioned in chapter 4.3.1.1, organizations have to define how to collect the usage data that are valuable and useful for organizations, which means organizations have to develop a tracking tool that could help to track and collect information about those usage data. There could be many possible ways to realize this tracking tool. Therefore, organizations at this level have to define the detailed possible solutions and define criteria to select solution.

  In order to satisfy the different tracking needs of different project, solutions for developing general tracking tools and specific tracking components for different projects should be separated.

- Developing detailed designs for the selected solutions

  Organizations at this level have to design the tracking tool components to meet the needs of collecting usage data.

  For general tracking tool development, organizations have to design general tracking tool components.

  As tracking needs may distinct from each other, organizations have to find out the specific tracking needs of each project, and develop its own customized tracking components, then add to the general tracking tool.

- Implementing the designs as a product

  After organizations developed the detailed designs for developing tracking tools, they firstly have to implement the design, and then develop support documentation: operator's manual and maintenance manual. They do not have to develop end-user training material, user's manual and online help as the end users of tracking tool do not have to use it. Instead, the tracking tool will proactively record the actions from users and store them as usage data.

Considering the general tracking tool and customized tracking components, for each project, they should have support documents for general tracking tool and customized tracking tool. Only the support documents for general tracking tool is the same for every project, whereas support documents for customized tracking component are different for each project.

- Evaluation of usage data

  The most important characteristic of level 3 is to develop tracking tools to check if users use software products in the same way as they designed to be. If there exist distinctions between users behavior towards software usage and expected software usage behavior, then software should give users instructions about how to proper use software. If users still not follow the instructions, organizations should request users to answer why they are not following the instructions provided, and based from those answers, organizations should make decision if they have to make change to their software products to make their products more fit with users' behavior.

4.3.1.3 Integrated Teaming

Suppose there are several developing teams in an organization, and each team is responsible for 1 or more projects. As we mentioned before, the tracking tool for each project is made up of two parts: the general tracking tool and customized tracking component. Although the general tracking tool could be developed by one project team, customized tracking components need the involvement of different project team member for each single project as the usage tracking needs are different.

So which people should be included in the team to develop tracking tool?

There are many possible ways to do this. I will just give two examples about how to set up a team for developing tracking tools for each project.

- Solution 1: Select a small group in each project team to design their own customized tracking components according to their specific needs, and then add the customized tracking component to the general tracking tool. There should be a separate group or person to responsible for designing the general tracking tool.

- Solution 2: Set up a team which includes people who design the general tracking tool and people who are responsible to collect the specific needs of each project, and to develop the customized tracking components for every project, and then integrate the general tracking tool with customized tracking components.

The difference of those two approaches is that solution 1 is more decentralization, as each project team design and develop their own tracking components, and then add them to general tracking tool, whereas solution 2 is more centralization, it only needs one person or a small group to design and develop customized tracking components for all projects, and then add them to general tracking tool for each project.

There could be many other ways to set up a team. It could be temporary or permanent. It is the organizations' choice to make. They could set up their team according to their specific needs.

4.3.1.4 Decision Analysis and Resolution

As we mentioned earlier, organizations have to figure out the usage requirements, so that they could design and develop tracking tool to collect information from users. Therefore, it is important to make decision which kind of usage data they want to track and which kind of usage data they do not want to track.

The following steps are showing how to make their decisions:

- Step 1: Establish the criteria for evaluating if usage data is valuable for them.
- Step 2: Identifying all types of usage data they are able to collect from users.
- Step 3: Evaluating those types of usage data by using the established criteria.
- Step 4: Selecting the types of usage data they want to collect.

4.3.1.5 Validation

In order to achieve level 3, organizations have to apply the standards on other similar projects to measure their usage data. The purpose of validation is to make sure the standards for measuring usage data is valid for other similar projects as well. Organizations at level 2 already have standards for measurement and analysis. But it still has its shortcomings. As the standards for measurement are based on only one project, if organizations want to apply the standards on other similar software projects, there might be mistakes. So what should we do if we want to apply standards on other similar software projects?

For similar projects, there are some functional similarities. They could even have similar tracking tools to collect usage data from users. For example, a company has a software product in different versions: windows version, mobile phone version and Mac OS version. The standards for measuring the usage data collected from those three different versions could be different. Therefore, we have to validate the usage standards we established at level 2 by analyzing usage data collect from users.

We have to collect usage data from users, and use the standards we established at level 2 to measure those usage data. Then we have to see how much deviation from the standards the result is. If the deviation is possible to be eliminated, then they have to find possible solutions to eliminate the deviation. If it is not possible by applying current technology or whatsoever, then they have to collect more usage from users more frequently, and alter the standards to the level which they could possibly achieve according to their historical usage data. By doing this, we could apply the usage standards on other similar projects. And help organizations to achieve level 3

4.3.2 Maturity level 3: Standardization

As we mentioned in level 2, every software application has their own usage tracking tool, this will bring organization some problems with the increasing software projects in an organization, such as exceeded budget to develop SU tacking tools. Therefore, something has to be done to avoid the chaos and reduce the cost of developing SU tools. By comparing those tools, organizations then could figure out the similarity of SU tools of each project, and come up with a general tracking tool. And this is the most important feature of maturity level 3: standardized tool. So organizations at this level should first define what data they want to collect, and what functions should be included in the general SU tracking tool, and then develop it. On the other hand, organizations at this level also provide flexibility of tracking tools for each software application. For each application, except from using General Tracking Tool (GTT), they could also develop their customized tracking component according to their specific needs that GTT could not satisfy.

4.3.3 Characteristics of level 3

- Defined standards for usage requirements development
- Defined technical solution for usage tracking
- Defined integrated teaming for usage tracking
- Decision Analysis for selecting valuable usage data
- Validation of usage standards for measurements

4.3.4 Life cycle of level 3

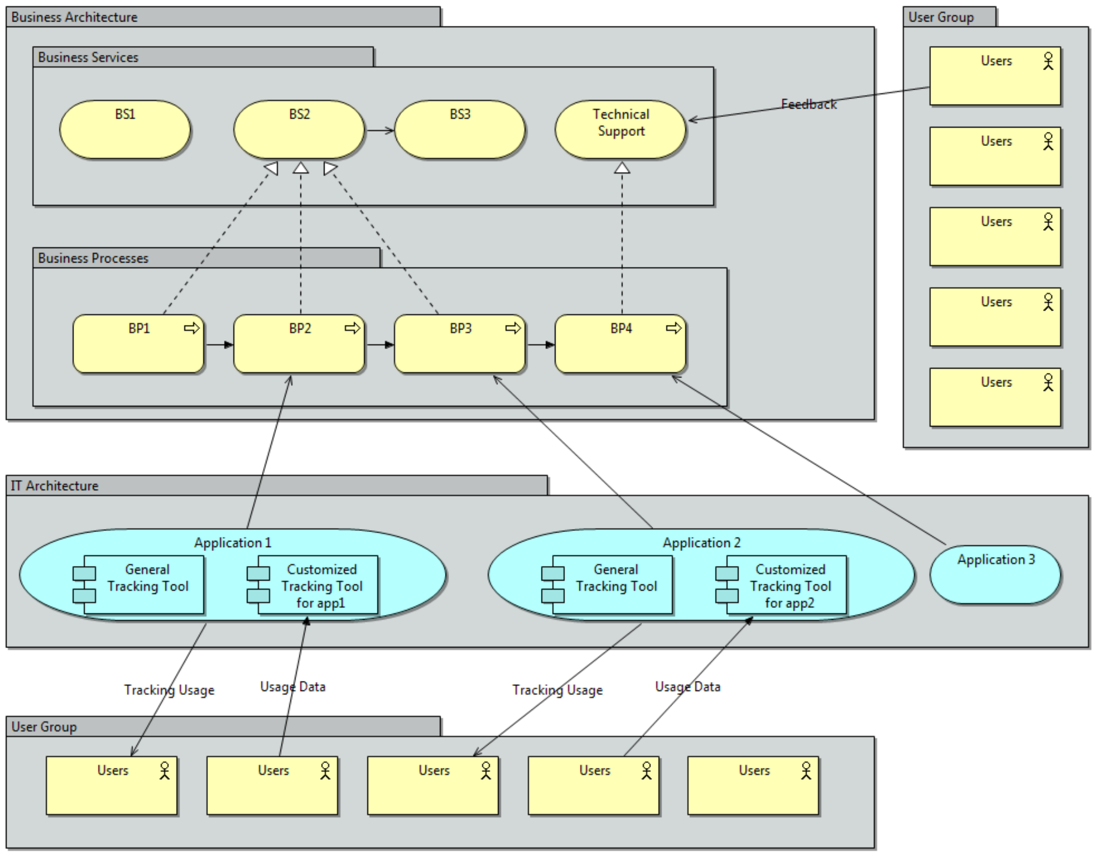


Figure 4-3-1: General Life Cycle of Maturity Level 3

There are two main changes from level 2 according to this diagram. One is the General Tracking Tool (GTT), and the other is Customized Tracking Tool (CTT) for each application. The GTT could be applied to every application in the organization to satisfy the most common tracking needs, whereas the CTT could give them flexibility to add their own tracking functions for each application. To develop the general tracking tool, organizations have to define what functions should be included by finding out the similarity of different tracking tools they had in level2, and then come up with the standard/common functions for GTT. After the tracking tool collect data from users, following remains the same: by analyzing tracking data, organizations then could make improvement for either their software engineering process or software applications.

Organizations at this level not only have standardized tracking tool but still provide enough flexibility for each application their specific needs of tracking functions. The benefits are obvious, they could have reduced complexity of IT systems, reduced IT costs. Otherwise, with the increasing amount of projects (applications), and each of them need to develop their own tracking tools; the development and maintain costs would be enormous. And the agility of IT architecture would be constrained, as each tracking tool has to be updated when some changes need to be done.

Let's take the online shopping website case for example:

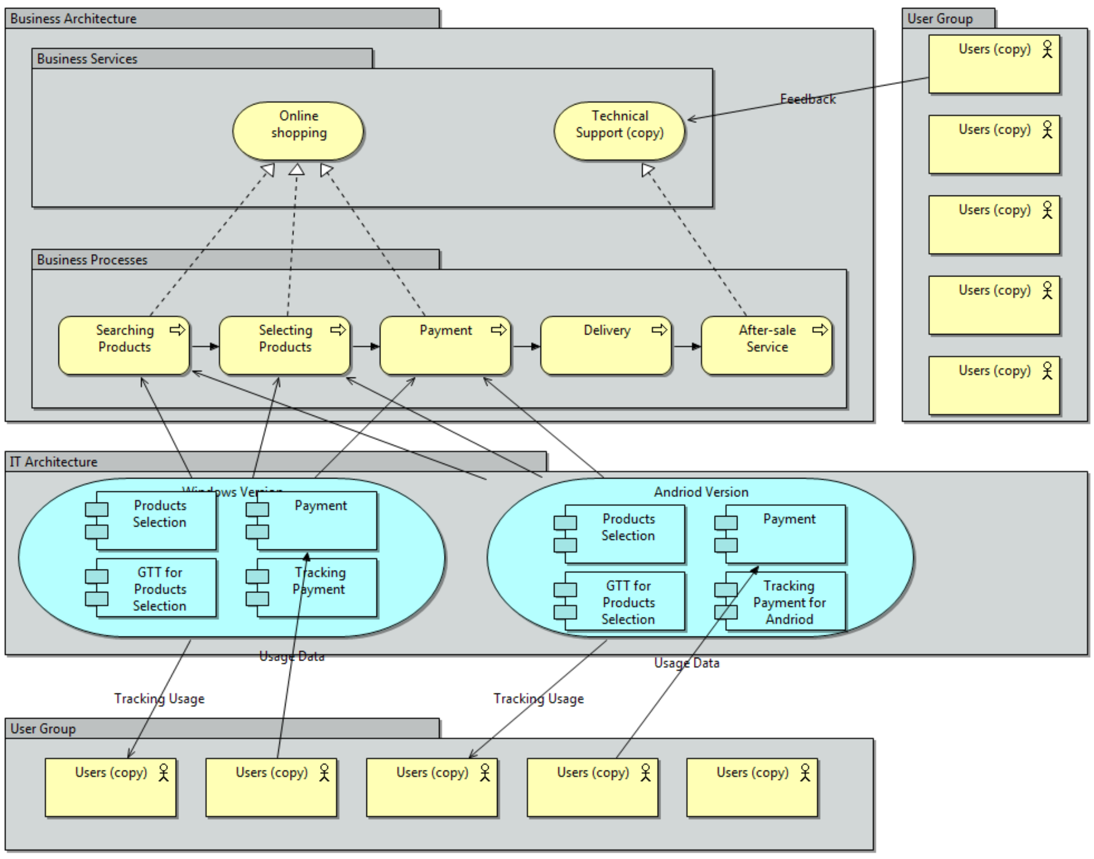


Figure 4-3-2: Example of Online Shopping Life Cycle of Maturity Level 3

Searching product, Selecting products and Payment are three processes supported by Windows Version application and Android Version application. And those two applications are similar projects which has its own tracking components/tracking tools. Instead of developing component for each application, organizations at this level have to analyze and find out the similarity of tracking components of Windows Version online shopping website and Andriod Version online shopping website, then developing the GTT. For example, tracking component for Application 1 and Application 2 both has a function to store the sequences of a successful transaction of users. Therefore this function should be added to the GTT component. And for each application, the specific tracking needs that the standardized tracking component could not satisfy, they could develop their own CTT to satisfy their specific needs. For example, there is a mobile phone application for Andriod Version online shopping website for payment process, which windows version does not have. Then for Andriod Version project, they could design their own CTT to track/monitor their specific payment processes.

4.4 Maturity Level 4

4.4.1 Origin of Maturity level 4

As we mentioned in chapter 3.5.3, the process areas of CMMI maturity level 4 are as follows:

- Organizational Process Performance
- Quantitative Project Management

The most important characteristic of CMMI maturity 4 is to identify special causes of process variation, and cope with the source of special causes to improve future handling. And those two processes mentioned above are defined to help identify the special causes of process variation. And those special causes of usage variation are the usage aspects we want to improve.

As for CMM for SU, the main target is also to identify special causes and eliminate those special causes. The difference is that while CMMI for SE focuses on identifying process variation, CMM for SU focuses on usage variation.

Before we discuss how to identify the special causes of software usage, there is a concept we have to clarify: usage aspect. There is no universal definition about what usage aspect is. Usage aspect could be understood as non-functional requirements or performance requirements of users' behavior towards software products that could be transferred to trackable functions or trackable data. "Easy to use" is an example of usage aspect. Let us see how this usage could be transferred to trackable data. Suppose a user want to use a specific function of a website. For example, he/she wants to find how much a delivery would cost before he/she places an order on the website. Then how many steps/actions has to proceed, and how many seconds or minutes will take until he find the delivery fee page from any other pages of the website is the trackable usage data we are talking about. "Conform to users' habit" is another example of usage aspect. Users' may have their own specific sequence to proceed a task when they are using the software product, which may be different from the inner sequence of this product. Then we have to track and store the sequence of proceeding the task. After this, we could find out the difference between those two sequences. Let's say S1 is the sequence of users' real usage sequence, and S2 is the inner sequence of software product. If we collect enough data from different users, and the result shows large part of users use S1 when they are proceeding the task. Then we have to change the working flow of this specific function of the software so that it could more conform to users' habit, therefore meeting their potential needs of the software product. Organizations should define their own usage aspects from historical usage data.

There is another concept we have to understand: direct usage data and indirect usage data. Direct usage data is the data directly representing users' single action towards using software products. We could also all it functional usage data. For example, if searching function is used or not is a direct usage data. For this data, we only have to collect data by monitoring the usage of searching function. Indirect usage data is the data represents users' sequential actions towards using software products. There may be several software functions involved. We could also call it as performance functional usage data. For example, the time users' spend on payment. There are many steps involved: selecting payment method, choosing bank, filling bank account information, and submitting payment request. Therefore, the time users' spend on payment processes is indirect data. We could not collect this kind of data by just tracking users' single action; instead, we have to design a separate function to collect this data.

As long as we understanding of the difference between direct usage data and indirect usage data, on the basis of this, for maturity level 4, we will discuss the process of identifying and eliminating special causes of indirect software usage data. There are two process areas involved: Identification and Solution of special causes.

4.4.1.1 Identification of Special Causes

For this process area, we firstly have to identify the indirect usage aspect needed to be improved. Therefore, organizations have to define the criteria to select a usage aspect. There are many possible ways to do this. For example, organizations could rank usage aspects according to the importance of them.

After selecting the usage aspect, organizations could either use historical tracking data they collected from users, or they could design a tracking component to track the usage data related to this usage aspect, and collect data to represent the usage aspect. It depends on whether or not historical tracking data could meet the needs of usage aspects on which organizations want to improve. About how to design and develop tracking components, see chapter 4.3.1. For example, as we mentioned earlier, if we want to improve the time users spend on payment, then we have to monitor how much time users spend on selecting payment method, filling bank account information, and submitting payment request. Then we could understand how many seconds or minutes users spend for this whole process.

The next step is to define what the period of collecting data is. After they tracked and collected data from users, organizations have to evaluate the data and find out special causes of usage data variation. The following steps will show us how to evaluate usage data collected.

- Steps 1: Formatting the data set.

The data set should be formatted so that we could analyze it in a systematically way

- Step 2: Selecting a method to represent the data

  There are many possible ways to represent the data. For example, we could use R function to draw a diagram to represent the mean of data, and deviation of each data point from the mean. By doing this, we will have a clear view of the distribution of data points.

- Step3: Identifying outliers

  For this step, we have to firstly define what the acceptable range of deviation from the mean of the data set is. Let's take the payment time for example again; if the mean of payment period is 10 seconds, then we have to define the range of acceptable payment period according to the diagram we have drawn at step2. Then they have to define the standards of evaluating if data points are outliers or not. For example, from the diagram, we define 60~100 seconds is the acceptable range of payment period, then those data points which value is less than 60 seconds or more than 100 seconds should be considered as outliers. And the outliers should be considered as the special causes of usage variation.

4.4.1.2 Solution for Eliminating Special Causes

After we identified the outliers, the next thing we have to do is to analyze the outliers, and find out solutions to eliminate those outliers.

Solutions for eliminating special causes should be based on users' behavior towards using software products as well. Therefore, organizations should define and develop a tracking component to monitor the usage aspect which has outliers according to evaluation standards, when the outliers occurred, the software product should provide support information and actions to collect information the reasons causing this, and request users to fill in their needs of this usage aspect. For example, as we mentioned earlier, the standard payment period is 60~100 seconds, let's say one user did not finish the payment, which means the payment period for him/she is infinite, which is longer than 100 seconds. This data should be considered as outlier. Then the payment period exceed 100 seconds, the software itself should pop out a form for this user to specify the reasons for the failure of this payment, and what are his/her suggestions about this payment process. If the reasons are software functions related, e.g. there are only limited payment method provided, and this user does not want to pay online, instead, he/she prefers payment after arrivals of goods. And that is the reason his/her payment period is outlier.

After we collect information about the possible reasons causing the outliers of usage data, we have to define possible solutions to eliminate the outliers.

There may be software functions need to be added to original software products, or there could be the needs of alteration of software functions. No matter what, organizations should implement the solutions on software product.

After implementing the solutions of eliminating outliers, organizations still have to monitor the usage aspect for a defined period, and make sure the outliers do not happen again.

4.4.1.3 Verification of Improvement

Once organizations identified special causes of usage variations, and also find out solutions to eliminate those variations. They have to track the usage aspect which has made improvement, to make sure the improvement has achieved expected goals.

If the improvement has proven to be a failure, then organizations have to come up with another solution for improvement, and after implementation of the improvement, organizations have to verify the improvement again until they find out a solution that could achieve expected goals.

On the other hand, if the improvement has achieved the expected goals, then organizations have to document the improvement processes and improvement details. Here are some points needed to be filed:

1. What are the possible reasons causing those usage variations?

2. What are the possible solutions to eliminate those special causes of usage variation?

3. What is the ultimate solution for improvement?

4. Why choose this ultimate solution over other solutions?

In this way, organizations could apply this improvement for similar projects. And they could also use this solution as a optical solution for their future projects.

4.4.1.4 Continuously Improvement of Usage Aspects

For organizations already managed to avoid special cause of usage deviation of one particular usage aspect. They may want to further improve this usage aspect. For example, as we mentioned in chapter 4.4.1.1, we want to make the entire payment period within 60~100 seconds, and we already achieved this target. There are no outliers exists, which means all usage data we collected after improvement meet this criteria.

Then what is continuous improvement? And what should we do?

Different from eliminating outliers, our target here is to reduce the variation from the standard. Take the payment period we mentioned before for example, the standard payment period is 50 seconds, and each payment is considered as a data point, what we should do here is to find out solutions to reduce the standard payment period to 20 seconds by applying new technology or adjustment of business processes or software products functions.

As we can see, the goal is somehow different, but the processes of making continuous improvement is mostly like what we have done for eliminating outliers: identification of causes; give solutions for improvement; verification of improvement and documentation.

4.4.2 Maturity level 4: Room for Improvement

The biggest characteristic of level 4 is that organizations at this level focus on usage improvement, which differentiates it from level 2 and 3.

For this maturity level, organizations' focus is on software usage improvement. The improvements are based on eliminating special causes of usage data variation, and make sure those variations will not occur in future projects.

4.4.3 Characteristics of level 4

For single project:

- One usage aspect has been defined for improvement
- Historical tracking data are used to help to find out solutions satisfying the needs for improvement
- Evaluation standards are established to assess the improvement
- Organizational standards of improvement are established

4.4.4 Life cycle of level 4

There are new IT components/processes/applications involved in the life cycle of maturity level 4, namely Data Extraction, Data Analysis and Improvements Generation. And those are the most importance characteristics of maturity level 4. Only based on the sequences of those IT applications/processes/components, organizations could deal with the special causes of usage data variation for single project. Therefore, in general, the life cycle of maturity level 4 is like this:

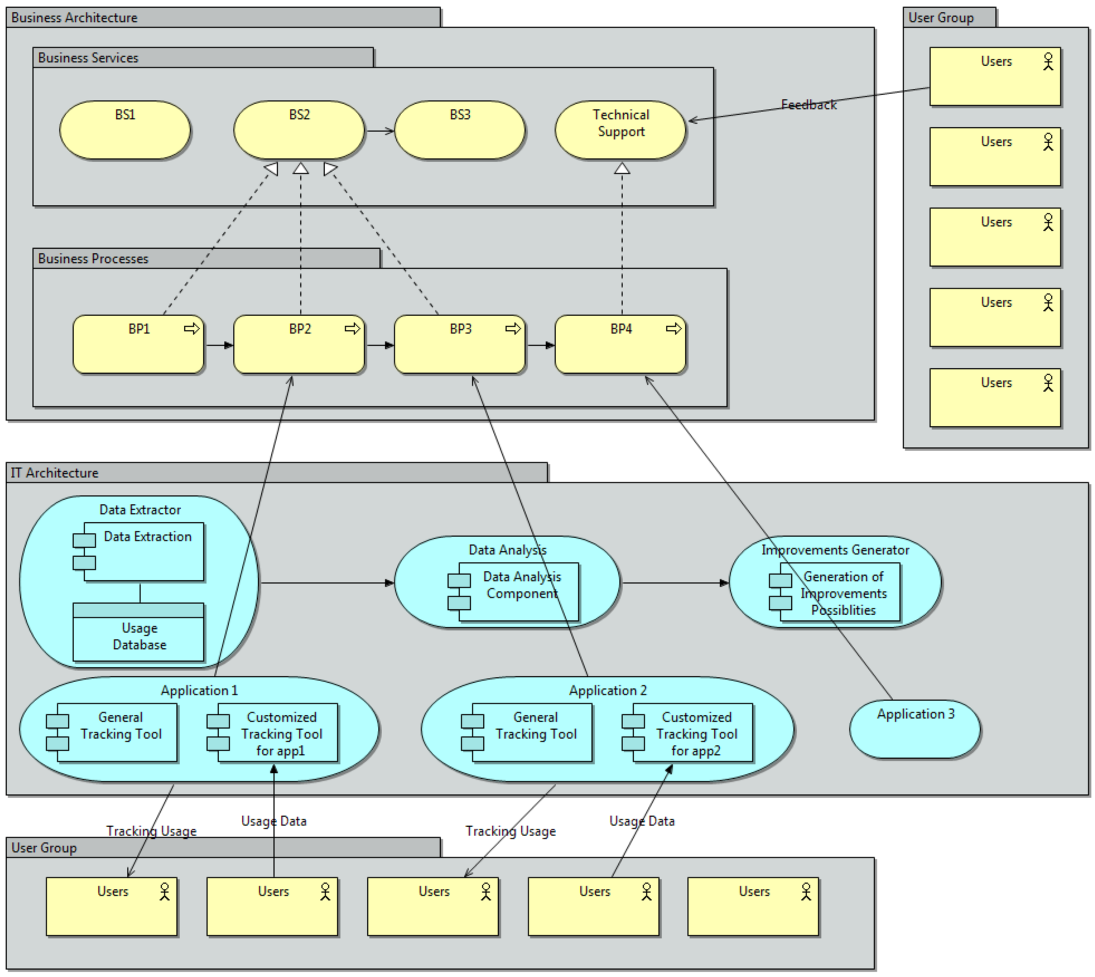


Figure 4-4-1: General Life Cycle of Maturity Level 4

The following example will show us the life cycle of organizations at maturity level 4.

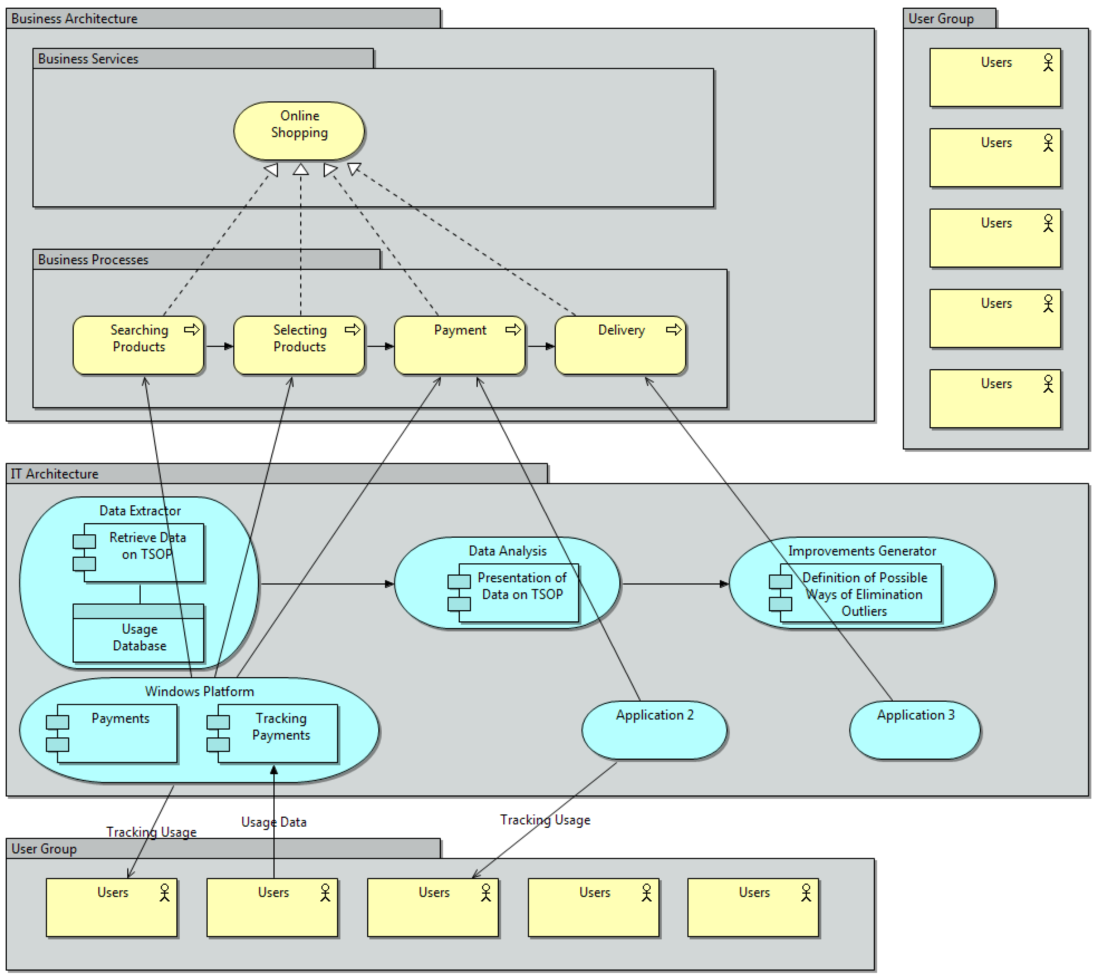


Figure 4-4-2: Example of Online Shopping Life Cycle of Maturity Level 4

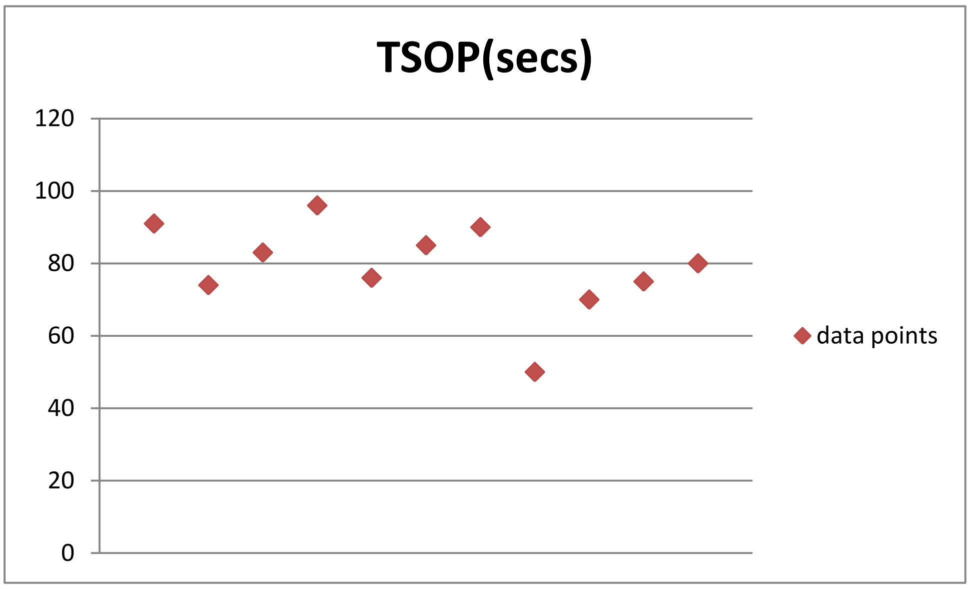
Suppose an online web shop wants to improve time period uses spend on payment. Here are the things they have to do:

1. Retrieve information needed for analysis, in this case, they have to retrieve payment period per user by, for example, data mining. If there does not exist data that could describe the information we need, then we have to find out solutions to get those data
2. Once retrieved data we needed for analysis, we have to analyze data.

Let's just take a look at a very simple example of analyzing tracking/usage data. Suppose the following data is usage data we collected and extracted from users.

	A	B	C	D	E	F	G	H	I	J	K
TSOP(secs)	100	120	130	150	200	101	110	50	70	108	103

A-K stands for different users from whom we tracked usage data.
TSOP means Time Spend on Payment for users.
Then we could use plot to present this usage data.

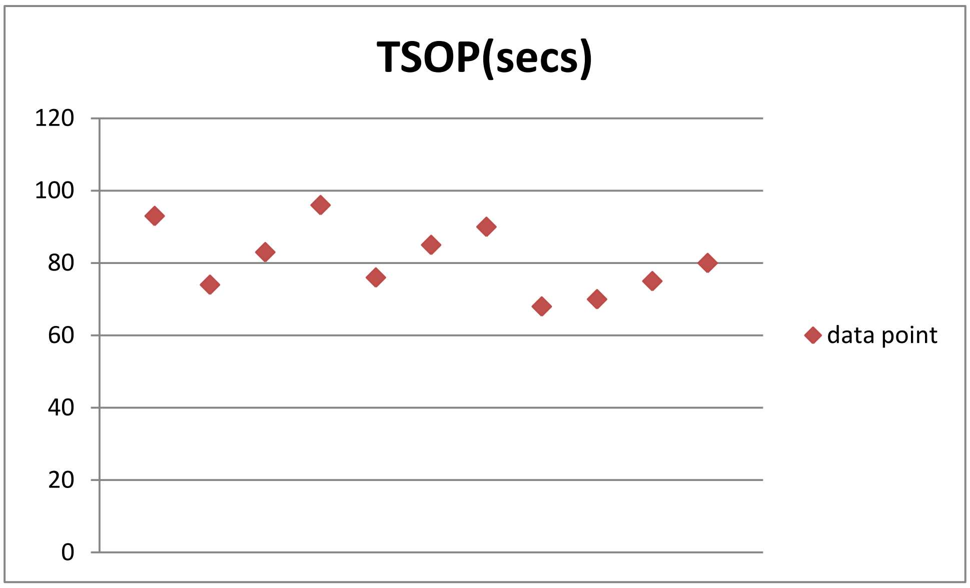


From the chart above, we could define outliers of this data set. For example, user E who spend 200 seconds, which has longer payment period than any other users. And user H only spend 50 seconds on payment, which is lower than other data points. So user E and user H are both considered as outliers.

3. After we defined outliers, the next step is to figure out what we have to do with outliers. Some outliers have positive effect towards payment period, e.g. data points H, whereas some outliers have negative effect. As our main target is to make improvement on payment period. We could either try to eliminate the outliers which has negative effects, or learn from outliers which has positive effects, or we could do both by comparing actions, times spend on each actions, etc. to find out possible causes of those differences. Organizations have to make their own decisions towards how to deal with outliers.
4. Once organizations have their solution to eliminate/learn from outliers, they have to make improvement on their product, and then tracking usage data for a period to check if they have achieved their goals or not. If not, they have to repeat step 2 and step 3 until they achieve their goals.

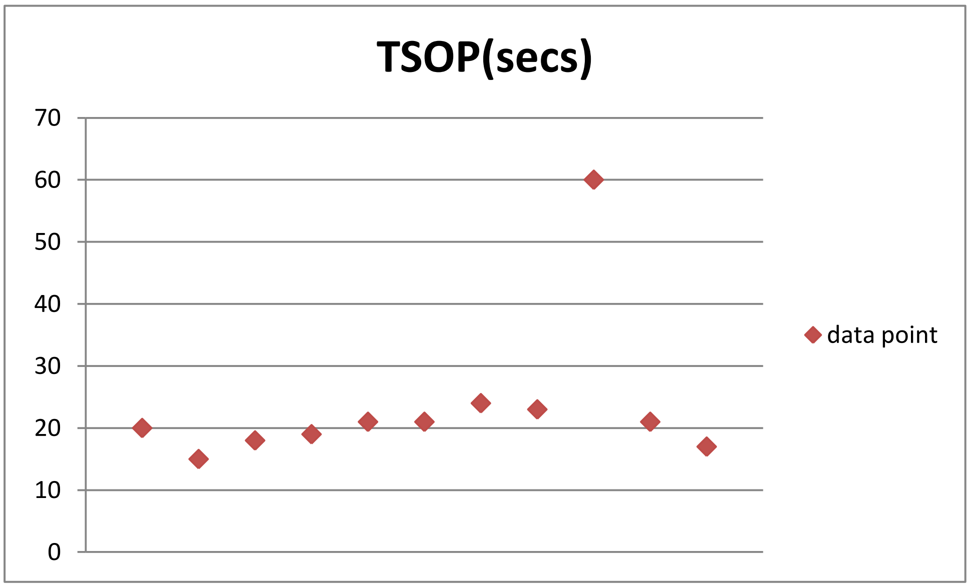
Suppose after organizations making improvement on their products/platform, they have already achieved its goals. So the following table presents time spend on payment after improvement.

	A	B	C	D	E	F	G	H	I	J	K
TSOP(secs)	93	74	83	96	76	85	90	68	70	75	80



5. After eliminating the outliers, continuous improvement can also be made. Suppose organization find out a solution to reduce the standard TSOP to 20 seconds.

	A	B	C	D	E	F	G	H	I	J	K
TSOP(secs)	20	15	18	19	21	21	24	23	60	21	17



As we can see from the chart above, most of data points are more or less equal to the standard TSOP. But still have one outlier, which is data point I. After we analyzed usage data, we decide this outlier is acceptable, thus this continues improvement is successfully implemented.

4.5 Maturity Level 5

4.5.1 Origin of Maturity level 5

The maturity level 5 process areas of CMMI are as follows:

- Organizational Innovation and Deployment
- Causal Analysis and Resolution

Maturity level 5 of CMMI is focusing on continuous improvement based on the understanding of common causes of variations inherent in processes. As for CMM for SU, the idea is similar. We focus on continuous improvement of software usage, which is based on the understanding of common causes of usage data variations. The difference between maturity 4 and 5 of CMM for SU is, we improve software usage based on the understanding of special causes of usage data variation at maturity level 4, whereas continuous improvement at maturity 5 based on the understanding of common causes of usage data variation. Therefore, at maturity level 5, we will focus on applying usage aspects improvement, which has already done at maturity level 4, on similar projects. For definition of similar projects, it could be functional similar, or it could be same functions but with different platforms/programming language, e.g. windows version/android vision/MacOS version.

4.5.1.1 Usage Aspects Management for Similar Projects

Once organizations succeeded on eliminating the outliers of one usage aspect for single project, the next thing they have to do is to apply this success on other similar projects in organizations.

Therefore, for this process area, our focus is standardization of usage aspect improvement. Chapter 4.4.1.1 and 4.4.1.2 already show us how to make software usage improvement based on eliminating special causes of usage data variation. So for this part, we have to establish standard processes to apply the success on other similar projects.

The following are the standards we have to set in order to achieve this goal:

- Usage aspects options

  The first thing organizations have to do is to check what usage aspects improvements are possible according to what they have achieved at maturity level 4. For this part, organizations have to list all possible usage aspects improvements.

- Usage aspects selection

  As organizations which have achieved maturity level 4 already defined usage aspects and have made improvements on those usage aspects for single project, what they have to do here is to decide which usage aspects they want to improve on other similar projects. So they have to set a standard for this selection processes. For example, they could establish some procedures or criteria to rank the improvement priority of different usage aspects.

4.5.1.2 Improvement Planning for Similar Projects

After organizations clearly know what improvement they want to apply on similar projects, the next thing they have to do is to make an improvement plan for this usage aspect.

The following things should be included in the improvement plan:

- Specify the usage aspect they want to improve for similar projects

  As usage aspects selection process has been done earlier (See Usage Aspects Management), the only thing they have to do for this part is documentation about which usage aspect they want to improve for similar projects

- Specify the reason they want to make this improvement

  Organizations have to specify and document the criteria and procedures to select usage aspects improvement.

- Specify the solution for this improvement

  As improvement on this specific usage aspect organization selected has been done for single project at maturity level 4. Organizations should have possible solutions for this improvement. Therefore, they only have to select the optimal solution which has successfully applied on single project. Organization should also establish a standard about under which circumstances they use previous solutions, and under which circumstances they have to come up with new solutions; and also how to select the most optimal solution. And this standard should be documented.

- Specify the implementation plan for the improvement

  After organizations have come up with their solution for applying usage aspect improvements on similar projects, they have to implement it. The implementation procedures

- Specify verification plan for the improvement

  After implementation, organizations have to verify if this solution/implementation has achieved the expected goal or not. Therefore organizations firstly have to define what the expected results are, following by establishing valuation standards for usage data or reusing the valuation standards which they have established at maturity level 4. The most important thing here is the standardization of verification criteria/procedures. In order to apply this verification process on other similar projects or future projects, those standards should be documented.

- Result Analysis

If the usage data they collected from users has achieved the evaluation standards, then the improvement is successfully done. If not, they have to find out the distinction between the usage data they collected from users and evaluation standards. And try to find out the reason causing those differences. At last, come up with solutions to eliminate the differences to achieve the evaluation standards.

Alternatively, they could try the backup solutions we mentioned earlier, and repeat those processes: performing the improvements, collecting data from users by using tacking tool, and then to evaluate the data by using evaluation standards.

4.5.2 Maturity level 5: Optimizing

The biggest characteristic of level 5 is that organizations at this level focus on usage improvement for similar projects, instead of single project which we did at maturity level 4.

For this maturity level, the improvements are focusing on common causes of usage data variation, and make sure those variations will not occur in similar project/future similar projects.

4.5.2 Characteristics of level 5

- Applying improvement of usage aspect on similar projects based on the understanding of common causes of usage data variation
- Standardization of procedures to apply usage aspects improvement on similar projects

4.5.3 Life cycle of level 5

IT architecture of maturity level 5 is almost the same as maturity level 4, only there are more similar applications are involved. So the architecture of maturity level 5 is like this:

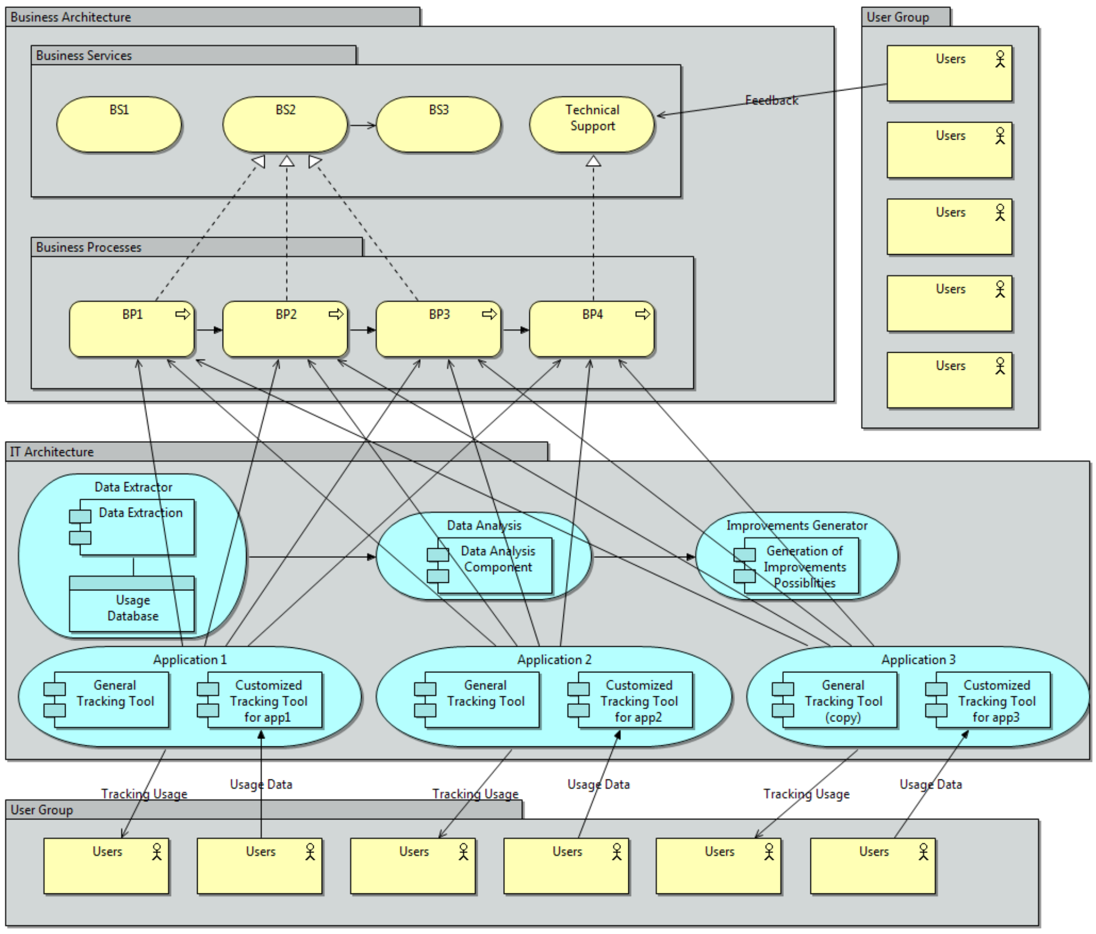


Figure 4-5-1: General Life Cycle of Maturity Level 5

Let’s take the online shopping website for example to demonstrate organizations at level 5. Suppose an organization has 3 different online shopping platforms available for users, namely Windows version, Android version, and MacOS version.

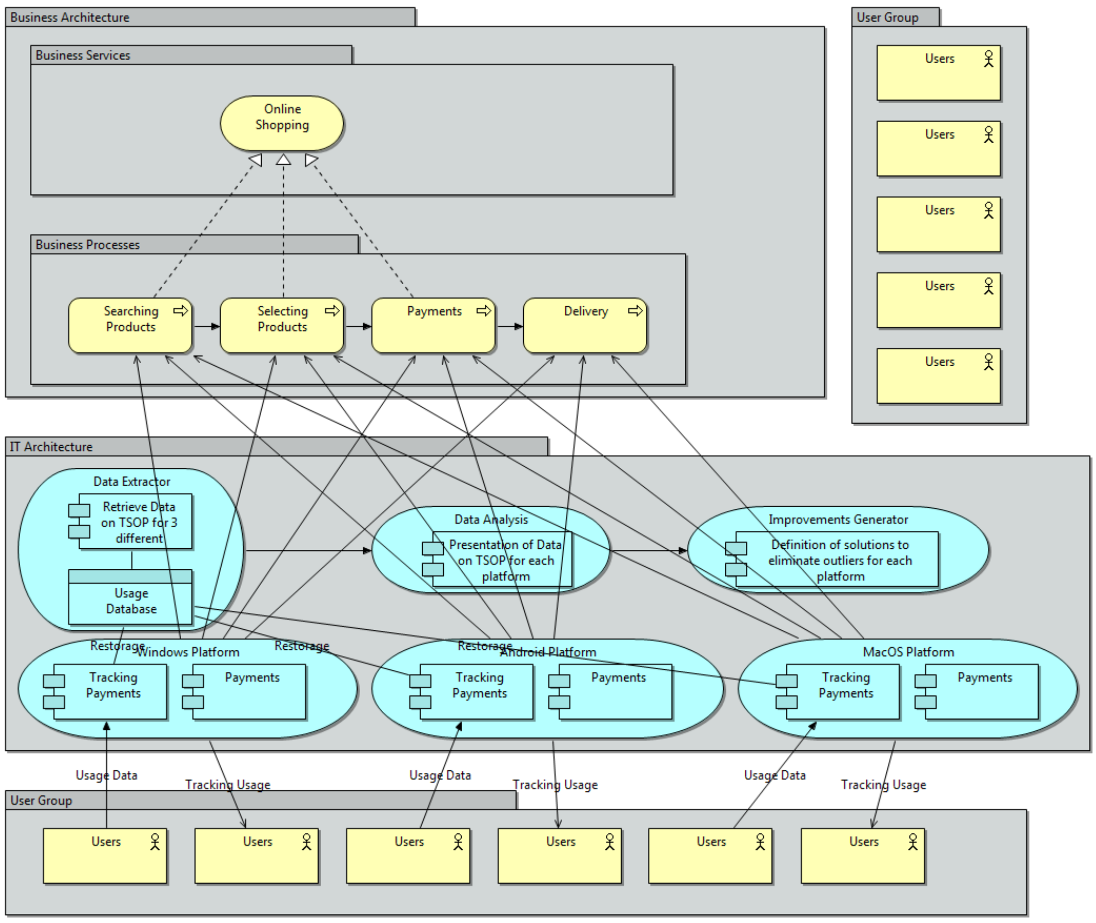


Figure 4-5-2: Example of Online Shopping Life Cycle of Maturity Level 5

Suppose this organization has already successfully implemented improvements on shopping period for windows platform, and they want to apply those improvements for Android platform and MacOS platform.

1. First of all, organization has to retrieve and present usage data from users or using historical usage data for Android platform and MacOS platform. The following tables are usage data collected from users.

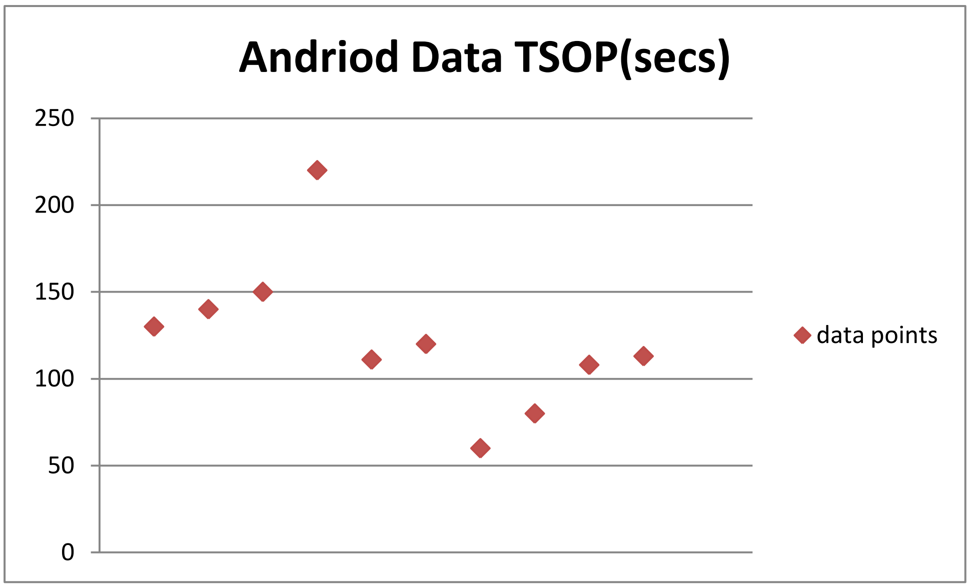
Data from Android platform:

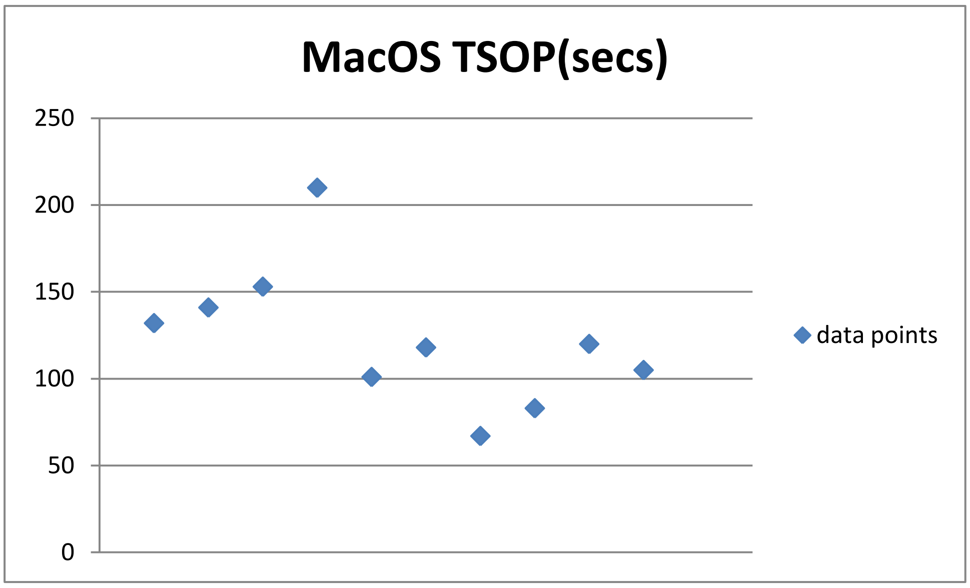
	A2	B2	C2	D2	E2	F2	G2	H2	I2	J2	K2
TSOP(secs)	120	130	140	150	220	111	120	60	80	108	113

Data from MacOS platform:

	A3	B3	C3	D3	E3	F3	G3	H3	I3	J3	K3
TSOP(secs)	125	132	141	153	210	101	118	67	83	120	105

A2-K2, A3-K3 stands for different users from whom we tracked usage data.
TSOP means Time Spend on Payment for users.
Then we could use plot to present two sets of usage data.



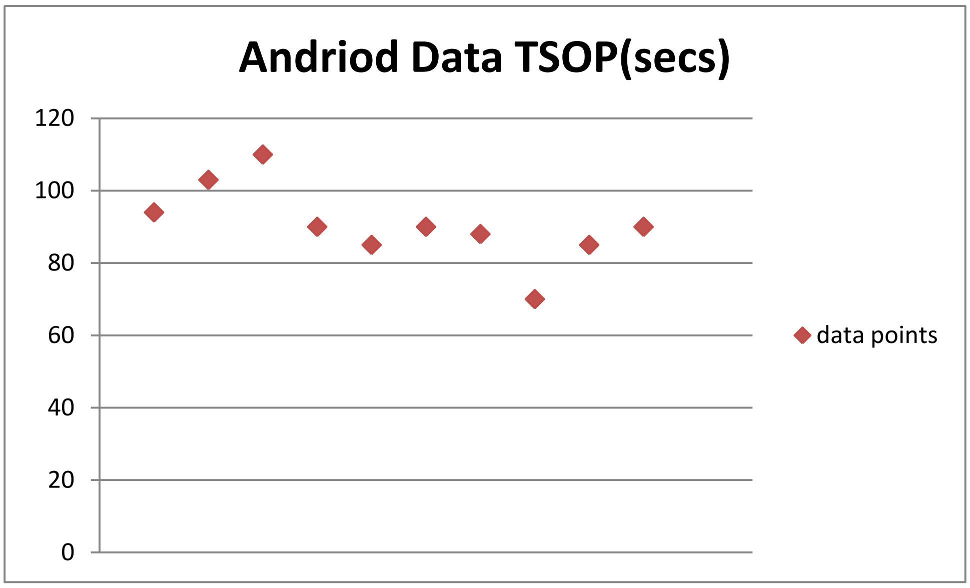


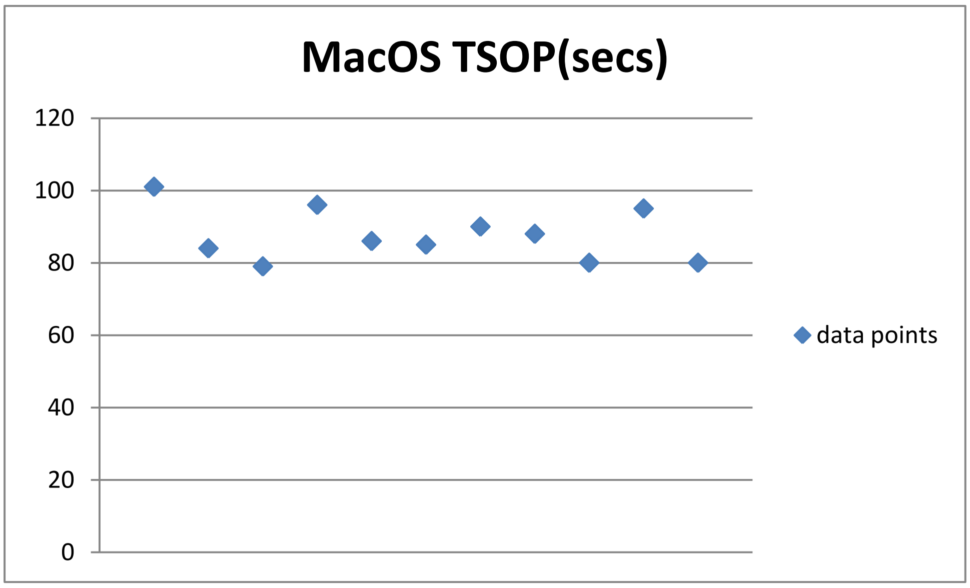
2. In order to eliminate outliers, e.g. data points D2 and D3, the next step is to apply improvement which has been successfully implemented on Windows platform on Android and MacOS platforms. Suppose organization decide to use the same solution as they did for Windows platform (see chapter 4.4.4); instead of finding out new solutions.

After the implementation of improvement:

	A2	B2	C2	D2	E2	F2	G2	H2	I2	J2	K2
TSOP(secs)	100	94	103	110	90	85	90	88	70	85	90

	A3	B3	C3	D3	E3	F3	G3	H3	I3	J3	K3
TSOP(secs)	101	84	79	96	86	85	90	88	80	95	80





We could see from the charts above, the implementation has successfully improved TSOP for other two platforms as well by applying the solution which has been done for Windows platform.

If this implementation turns out to be failed, then organizations have to follow what they have done for maturity level 4, which is also documented at maturity level 5(see chapter 4.5.1.2) and find out solutions and implement until they successfully improve the usage aspects.

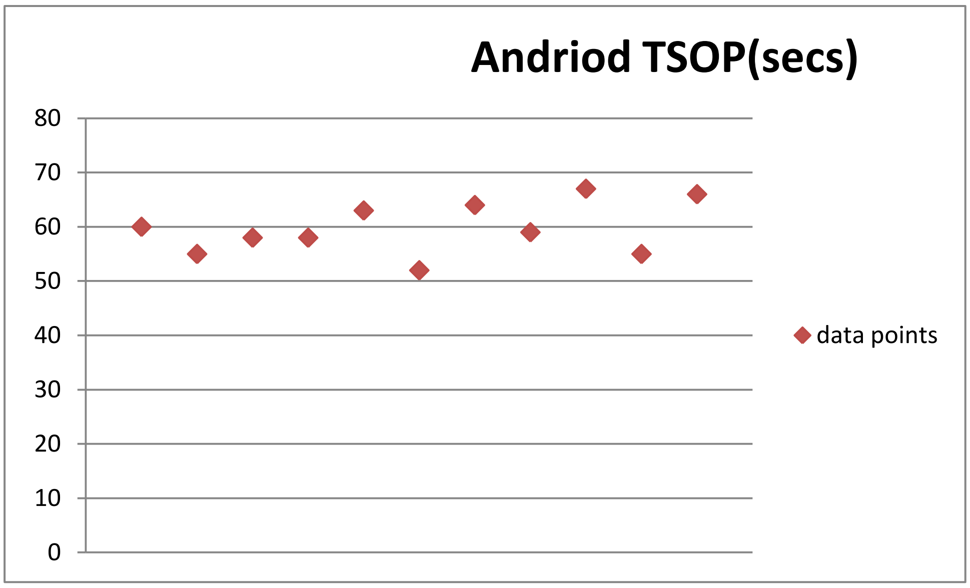
3. For continuous improvement, organizations only have to repeat the same steps as we mentioned above, which is applying the successful improvement of Windows platform on Android and MacOS platforms.

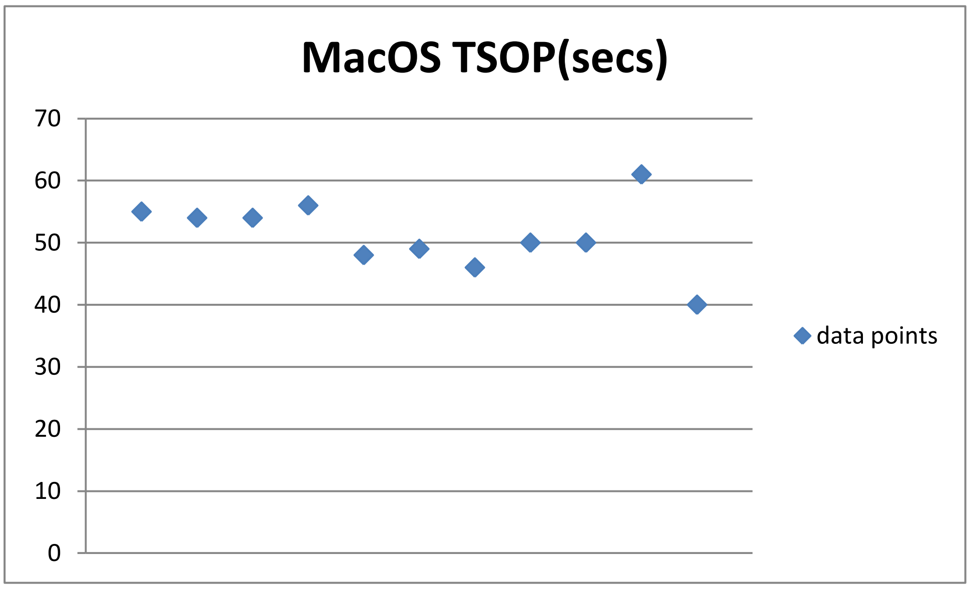
Suppose organization wanted to reduce average TSOP to 60 seconds for Android platform and 50 seconds for MacOS platform. They could choose to use the same technology/solutions which they have been successfully applied for Windows platform.

After implementation of improvements:

	A2	B2	C2	D2	E2	F2	G2	H2	I2	J2	K2
TSOP(secs)	60	55	58	58	63	52	64	59	67	55	66

	A3	B3	C3	D3	E3	F3	G3	H3	I3	J3	K3
TSOP(secs)	55	54	54	56	48	49	46	50	50	61	40





As we can see from the chart above, by applying solutions which has been successfully implemented at maturity level 4 on other similar projects, which is

Android and MacOS platform in this case, they also made a successful improvement of this specific usage aspect. And that’s the main target of Maturity level 5.

When organizations want to make improvement on more usage aspects, they only have to repeat the iteration of those process areas at maturity level 4 and 5.

Chapter 5: Comparison of CMMI model

From Chapter3, we know CMM for SU origins from CMMI for SE model. So they are similar in essence. For maturity level 1, everything is chaotic, and then things become more systematic at maturity level 2. At level 3, organizations' main focus is on standardization, and for maturity level 4, organizations start to switch their focus to special causes of variation. For the last stage --maturity level 5, organizations focus on common causes of variation.

Although those two models share the same idea, they still have many differences in detail. The form below shows how those two models are different from each other.

	CMMI for SE	**CMM for SU**
Focus	Process improvement	Usage improvement
Industry	Software development	Not specific
Improvement	Inherent	Inherent and External
Goal	Meet business goals of an organization	Finding out users' real requirements
Users' Requirements	Specific, relatively stable	Need to be discovered, unstable
Process Areas	Detailed	General

First of all, those two models are focusing on different aspects. For CMMI model, organizations' focus is project management and process improvement. But for CMM for SU, organizations' focus usage improvement by observing/tracking users' behavior. Users' behavior is the most critical point of this model. How can we find out users' real needs? Are users' satisfied with our products? To what extend users' like our products? All those kind of questions could not be answered by users. They may have a feeling about it, but they may not express it in a rigorous way that we could understand. However, by observing/tracking their behavior of using products, we could understand it without ambiguity. As a matter of this, the goals of different models are not the same. The goal of CMMI is to meet business goals of an organization, while the goal of CMM for SU is to find out users' real requirements by observing/tracking users' behavior toward software usage.

Secondly, the industry those two models could be applied to are also different. CMMI for SE mostly applied on software engineering industry, as for

CMM for SU, we also start from software industry, but then we could extend those models for other industry as well. This is mainly because usage is not exclusive for software; other non-software products also have its usage aspects. As long as usage improvement achieved its maturity in software industry, we could apply this model on other industry as well.

A significant different of those two models are the involvement of outside resources. For CMMI model, organizations improvements are inherent. They define different specific and general goals, and process areas that could help them to achieve process improvement within the organization. But for CMM for SU, organizations' improvements are both inherent and external. Organizations need to have a closely-related relationship with their customers, observing/tracking users' behavior, collecting information, and based on this, organizations could analysis information collected from users and then have a better understanding of users' real needs. Thus, organizations could improve their software usage.

Users' requirements are different for two models. As for CMMI model, the users' requirements are quite clear and standardized. Organizations have their procedures or standards to translate users' requirements into a document that could avoid ambiguity, no matter what method they are using to achieve this. But in CMM for SU model, users' requirements are need to be discovered, not from what they are telling you they want, but what they really want via their behavior of using software.

Last but not least, the process areas of two models are different. As for CMMI for SE, process areas are more detailed and explained by showing concrete examples, and they covered almost all aspects of software engineering processes. For CMM for SU, process areas are more general, we only defined what process areas organizations should focus for each maturity level, but we have not defined how to do it in detail or with concrete example. For example, we defined that organizations should have their own way to track software usage, but we do not define what tools organization should use to track the usage.

Chapter 6: Conclusion

CMM for SU has five maturity levels, namely Ad hoc, Basic Supporting, Standardization, Room for Improvement, and Continuous Improvement.

At maturity level 1, organizations have no common understanding of what software usage is and what software usage could help them improve their products.

At maturity level 2, organizations have knowledge about the importance of software usage, and they start supporting the usage of their customers on single project by defining the following process areas:

- Usage Requirements and their Management
- Usage Planning
- Usage Monitoring
- Users Agreement Management
- Measurement and Analysis

At maturity level 3, organizations start to standardize their software usage by defining the process areas, so they could support the usage on similar projects, not only one project:

- Usage Requirements Development
- Technical Solution
- Integrated Teaming
- Decision Analysis and Resolution
- Validation

At maturity level 4, organizations shift their focus to special causes of software usage variation, in order to improve software usage for single project:

- Identification of Special Causes
- Solution for Eliminating Special Causes
- Verification of Improvement
- Continuously Improvement of Usage Aspects

At maturity level 5, organizations' target is to improve software usage for similar projects:

- Usage Aspects Management for Similar Projects
- Improvement Planning for Similar Projects

ArchiMate modeling tool—Archi also helped a lot for building this model. It not only helped to describe the life cycle of each maturity level, but also gave me insight about what processes should be included in each maturity level. Besides, Archi also gave me confidence to explain each maturity level by building a very clear framework. It also helped me to create appropriate examples for each maturity level in order to explain how to achieve each maturity level.

Chapter 7: Future Research

For this maturity model, we only generally defined process areas for each maturity level. There are many possibilities to extent this topic. As far as I see, there could be at least three possible directions for future research.

The first direction would be adding complementary contents. This means there will be more process areas be added for each maturity level, and more detailed definition/explaining/example will be added for existent process areas or new process areas.

The second direction would be the structural change of maturity levels. With the development of new technology or people's knowledge about software usage or whatsoever, some maturity levels are too easy to achieve in the future. Therefore, we have to re-construct the maturity levels. For example, process areas at maturity level 3 now is too easy to achieve in the future, therefore, we should set a higher standards for organizations to achieve, may be some process areas in maturity 4 should be moved to maturity level 3 in the future.

Direction three will be the extension of model to other industries. When this model is proved to be a successful model for software usage, then we could try to find out if this model could be applied to other industries as well, and find out how to apply this model on other industries.

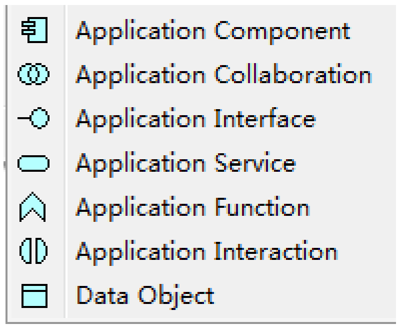
Appendix 1: Terminology and Abbreviation

1. CMMI: capability maturity model integration
2. SE: software engineering
3. SU: software usage
4. TSOP: time spend on payment

Appendix 2: Introduction to Archi

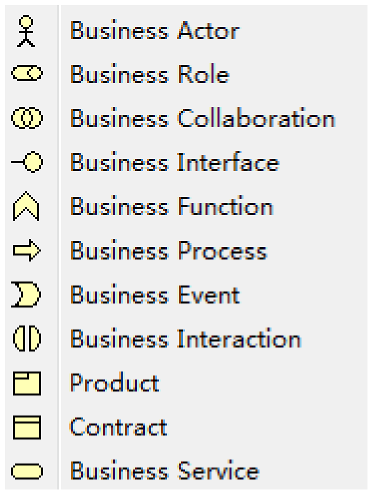
Archi is a free, open source, cross-platform tool and editor to create ArchiMate models. We will only briefly introduce the elements and relations which will be used for representing life cycle of each maturity level for software usage. If you are interested in this tool, you could visit their website for more information: http://archi.cetis.ac.uk/.

Elements for IT Layer:



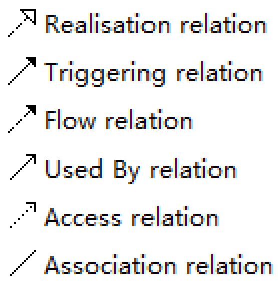
Application Component
Application Collaboration
Application Interface
Application Service
Application Function
Application Interaction
Data Object

Elements for Business Layer:



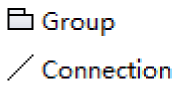
Business Actor
Business Role
Business Collaboration
Business Interface
Business Function
Business Process
Business Event
Business Interaction
Product
Contract
Business Service

Relations between Elements:



Realisation relation
Triggering relation
Flow relation
Used By relation
Access relation
Association relation

Groups and Connection:



Group
Connection

Reference

[1].
http://en.wikipedia.org/wiki/CMMI#Maturity_levels_in_CMMI_for_development

[2]. Page 10, http://www.sei.cmu.edu/reports/02tr012.pdf

[3]. Page 10, http://www.sei.cmu.edu/reports/02tr012.pdf

[4]. Page 10-11 http://www.sei.cmu.edu/reports/02tr012.pdf.

[5]. Page 11-14, http://www.sei.cmu.edu/reports/02tr012.pdf

[6]. Page 47-65, http://www.sei.cmu.edu/reports/02tr012.pdf

[7]. Page 81-556, http://www.sei.cmu.edu/reports/02tr012.pdf