GUIDELINE

Use Case Points Estimation

Code	**03be-HD/PM/HDCV/FSOFT**
Issue/revision	**2.0**
Effective date	**15-Oct-2005**

TABLE OF CONTENT

1 ESTIMATING WITH USE CASES - THE USE CASE POINTS APPROACH

The Use Case Points Approach, from Rational Software, is a modification of the Function Points method of estimation. As the process maturity of FPT increases, there should be attempts to tailor this estimation method to include or exclude any features specific to FPT.

The steps are explained below. However to implement the steps, the EstimateUCP.xls Excel spreadsheet can make computation simple – Estimate Using UCP Approach

1.1 Estimating Size

1.1.1 Step 1: Weighting Actors

The process starts by considering the actors in the system. For each actor, determine whether it is simple, average or complex. A simple actor represents another system with a defined Application Programming Interface (API). An average actor is either another system that interacts through a protocol such as TCP/IP, a person interacting through a text-based interface, or data store such as files and RDBMS. A complex actor is a person interacting through a graphical user interface.

Count how many of each kind of actors are present in the system. Multiply by weighting factor. Add these products together to get a total.

Actor Type	Description	Factor
Simple	Program interface	1
Average	Interactive, or protocol driven interface	2
Complex	Graphical interface	3

1.1.2 Step 2: Weighting Use Case

A similar process is followed for use cases. Used use cases or extending use cases do not need to be considered. For each use case, determine whether it is simple, average, or complex. The basis of this decision is the number of transactions in a use case, including secondary scenarios. For this purpose, a transaction is defined to be an atomic set of activities, which is either performed entirely or not at all. A simple use case has 3 or fewer transactions, an average use case has 4 to 7 transactions, and a complex use case has more than 7 transactions.

If analysis classes have been defined for the system and it has also been identified as to which ones are used to implement a particular use case, use this information in place of transactions to determine the use case complexity.

By Use Case Description

Use Case Type	Description	Factor
Simple	3 or fewer transactions	5
Average	4-7 transactions	10
Complex	>7 transactions	15

By Analysis Classes

Use Case Type	Description	Factor
Simple	Fewer than 5 analysis classes	5
Average	5-10 analysis classes	10
Complex	More than 10 analysis classes	15

Count how many of each kind of use case are present. Then, multiply each type by the weighting factor specified in the given table. Add these products to get a total.

Add the total for actors to the total for use cases to get the **unadjusted use case points (UUCP)**. This raw number will be adjusted to reflect the project's complexity and experience of the people on the project for effort estimating.

Estimator may refer to Appendix 2.1 for some tips, guidelines on resolving actors and Use Case structure matters.

1.1.3 Step 3: Weighting Technical Factors

Start by calculating the technical complexity of the project. This is called the technical complexity factor (TCF). To calculate the TCF, go through the following table and rate each factor from 0 to 5. A rating of 0 means the factor is irrelevant for this project, 5 means it is essential. Now, for each factor multiply its rating by its weight from the table. Finally add together all these numbers to get the total T factors.

– TFactor = ∑(Tlevel) * (Weighting Factor)

– TCF = 0.6 + (0.01 * TFactor)

Technical Factors for System and Weights

Factor	Factor Description	Weight
T1	Distributed system	2
T2	Response or throughput performance objectives	1
T3	End-user efficiency (online)	1
T4	Complex internal processing	1
T5	Code must be reusable	1
T6	Easy to install	0.5
T7	Easy to use	0.5
T8	Portable	2
T9	Easy to change	1
T10	Concurrent	1
T11	Includes special security features	1
T12	Provides direct access for third parties	1
T13	Special user training facilities required	1

Size of software calculated by Use Case Point is

SzUC = UUCP * TCF

Estimator identifies off-the-shelf or reusable software components. The size of off-the-shelf or reusable software components should be subtracted from the size of software.

Estimator may refer to Appendix 2.2 for rating technical factors guidelines.

1.2 Estimating Effort

1.2.1 Step 4: Weighting Environmental Factors

Consider the experience level of the people on the project. This is called the environmental factor (EF). To calculate EF, go through the table below and rate each factor from 0 to 5. For factors F1-F4, 0 means no experience in the subject, 5 means expert, and 3 means average. For F5, 0 means no motivation on the project, 5 means high motivation, and 3 means average. For F6, 0 means extremely unstable requirements, 5 means unchanging requirements, 3 mean average. For F7, 0 means no part-time technical staff, 5 means all part-time staff, 3 means average. For F8, 0 means easy–to-use programming language, 5 means very difficult programming language, 3 means average.

For each factor, multiply its rating by its weight from the table given below. Finally, add all the numbers together to get the total E Factors.

- EFactor = ∑(Flevel) * (Weighting Factor)
- EF = 1.4 + (-0.03 * EFactor)

Environmental Factors for Team and Weights

Factor	Factor Description	Weight
F1	Familiar with FPT software process	1.5
F2	Application experience	0.5
F3	Paradigm (e.g. Object–oriented) experience	1
F4	Lead analyst capability	0.5
F5	Motivation	1
F6	Stable requirements	2
F7	Part-time workers	-1
F8	Difficult programming language	-1

Estimator may refer to Appendix 2.3 for rating environment factors guidelines.

1.2.2 Step 5: Use Case Points

Calculate Use Case Points (UCP) as below:

UCP = UUCP * TCF* EF

1.2.3 Step 6: Effort Estimate

UCP method suggests usage of 20 person-hours per UCP for a project estimate. But a close examination of the data suggests a refinement based on experiences. Count how many factor ratings of F1-F6 are below 3 and how many factor ratings of F7-F8 are above 3. If the total is 2 or less, use 20 person-hours per UCP. If the total is 3 or 4, use 28 person-hours per UCP. If the total is 5 or more, try to make changes to the project so the numbers can be adjusted. The risk of failure is quite high otherwise.

1.2.4 Step 7: Apply the UCP estimation model to specific project context

Specific project context requires the adjustment of calculated effort. This is called the project factor (PF). There are two types of project context factors: project life cycle coverage factors and project specific assumption factors.

Apply project life cycle coverage factors (P0 – P5)

To apply P0-P5, go through the table below and rate each factor. For factor P0, D means that the type of project is development, M means maintenance and P means Pilot. For factors P1-P5, 5 means full activities of the process and all work products of the process have to be created, 0 - No, 3 - Average. *Note: P0 will impact to the weight of all factors P1-P5.*

Factor	Factor Description	Weight (D)	Weight (M)	Weight (P)
P0	Project type	N/A	N/A	N/A
P1	Requirement	0.12	0.09	0.15
P2	Design	0.16	0.09	0.15
P3	Code	0.48	0.4	0.4
P4	Test	0.18	0.24	0.2
P5	Deployment	0.06	0.18	0.1

For each factor (except P0), multiply its rating by its weight. Finally, add all the numbers together to get the total PFactorL and calculate the impact of project life cycle coverage factors PFl

- PFactorL = ∑(Plevel) * (Weighting Factors)
- PFl = 0.2 * PFactorL

Apply project specific assumption factors (P6 – P9)

To apply P6-P9, go through the table below and rate each factor. For P6, 0 means 1 staff onsite with full time onsite in the project, 5 - No, 3 - Average. For P7, 5 means high risk, 0 - No, 3 - Moderate. For P8, 5 means a lot of documents required to be translated, 0 – No, 3 - Partly. For P9, 0 means active participation of customers in the project, 5 - No, 3 - Partly.

Factor	Factor Description	Weight
P6	Onsite	0.4
P7	Risk	0.12
P8	Translation	0.6

Factor	Factor Description	Weight
P9	Customers	0.4

For each factor, multiply its rating by its weight. Finally, add all the numbers together to get the total PFactorS and calculate the impact of project specific assumption factors PFs

- PFactorS = $\sum$(Plevel) * (Weighting Factor)
- PFs = 1 + (0.05 * PfactorS)

Calculate the final estimation result

To produce the final estimation, multiply the result from step 6 with PFl and PFs.

2 APPENDIX

2.1 Actors and Use Case structure matters

Generalization between actors and structure of Use Case matters a lot to the estimation. Estimator should consider the following aspects:

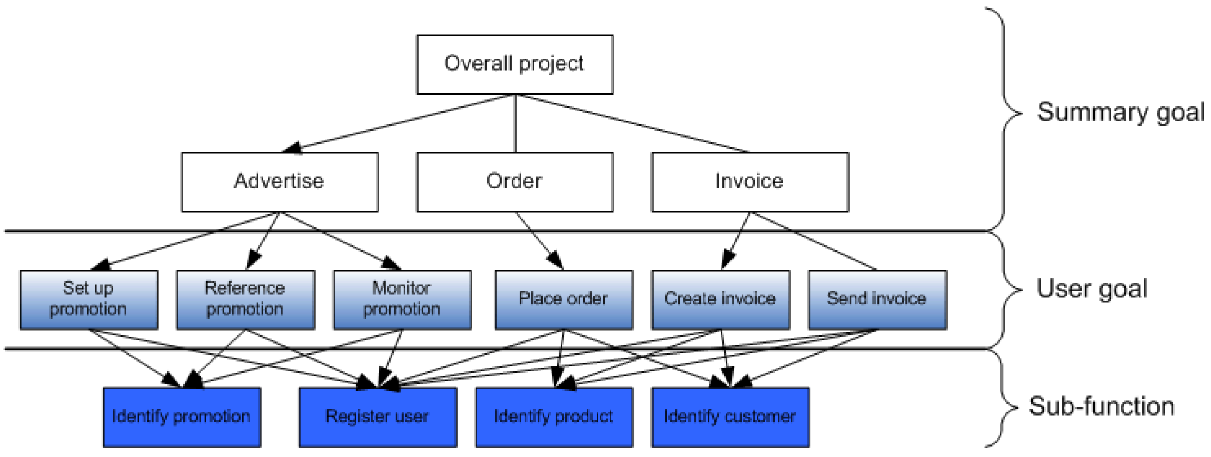
2.1.1 Generalization between actors

If two actors have 80% in common, put them in generalization relationship and count them as only one.

For example, with a CMS (Content Management System), we may have an actor called Publisher who is responsible/ able to compare versions of a content object, approve a content object for publishing and rollback a released content object back to an older version. And we also may have another actor called Manager who has all rights of the Publisher actor, however with a more right to generate/view reports. These two actors should be generalized into a *super-actor* and counted as only one when estimating.

2.1.2 Use Case goal levels

Use Cases have three named goal levels: summary goal, user goal and sub-function level. In most circumstances, user goal levels are completed by one person, in one sitting (1 to 20 minutes). Summary goal involves multiple user goals; runs over hours, days or years. Sub-function goals are required to carry out user goals. See below for an illustration:



User goal level is the right level of Use Case on which estimation can be done. A user goal level Use Case can be mapped to an elementary process of Function Points.

2.1.3 The level of detail in Use Case

The level of detail of Use Case impacts the estimation a lot; it depends on the estimator as well as the Use Case template he/she used. The estimator should count only transaction steps which add business value for the estimation.

2.1.4 Splitting and merging Use Cases

If two Use Cases have 60% scenario in common, accommodate them in one Use Case using alternative scenario. For instance, CRUD (Create, Read, Update, and Delete) should be wrapped up in one or two Use Cases if they do not have different business validations.

2.1.5 Included and extension Use Cases

Some researchers recommend that these Use Cases should not be counted, meanwhile some recommend the inversion. The estimator should pay attention to the included and extension Use Cases that have essential functionalities and estimate them with WBS. The result of WBS should be compared with UCP to produce an accurate estimation.

2.2 Guidelines for rating technical factors

2.2.1 Distributed system

Description: Is the system having distributed architecture or centralized architecture?

Rating:

Rating	Description
0	Application does not aid the transfer of data or processing function between components of the system
1	Application prepares data for end-user processing on another component of the system such as spreadsheets and RDBMS (e.g. common Web applications)
2	Data is prepared for transfer, then is transferred and processed on another component of the system, but not end-user processing (e.g. Web Services)

Rating	Description
3	Distributed processing and data transfer are online and in one direction only (e.g. Distributed problem solving system with a component acting as a coordinator/moderator to divide the overall problem into a series of sub-problems and to assign sub-problems to associated components, low interaction among components)
4	Distributed processing and data transfer are online and in both direction (e.g. Distributed problem solving system with a component acting as a coordinator/moderator to divide the overall problem into a series of sub-problems and to assign sub-problems to associated components, high interaction among components to share knowledge, processing result, negotiate, etc.)
5	Processing functions are dynamically performed on the most appropriate component of the system (e.g. Distributed problem solving system without a dedicated co-ordinator, the division of the overall problem and assignment sub-problems are performed by all associated components or by any component, Mobile Intelligent Agent system)

2.2.2 Response or throughput performance objectives

Description: Does the client need the system to fast? Is time response one of the important criteria?

Rating:

Rating	Description
0	No special performance requirement were stated by the user
1	Performance and design were stated and reviewed but no special actions were required
2	Response time or throughput is crucial during peak hours. Feasibility can be established with minimal design effort
3	Response time or throughput is crucial during all business hours. Processing deadline requirements with interfacing system are constraining
4	In addition, stated user performance requirements are stringent enough to require performance analysis tasks in design phase
5	In addition, performance analysis tools were used in design, development, and/or implementation phases to meet the stated user performance requirements

2.2.3 End-user efficiency (online)

Description: Was the system designed for end-user efficiency?

Rating factors: There are 16 factors which govern how this point is rated

#	End-user efficiency factor
1	Navigation aids (for example, function keys, jumps, dynamically generated menus)
2	Menus
3	Online help and documents
4	Automated cursor movement
5	Scrolling
6	Remote printing (via online transaction)
7	Re-assigned function keys
8	Batch jobs submitted from online transactions
9	Cursor selection of screen data
10	Heavy use of reverse video, highlighting, colours underlining, and other indicators
11	Hard copy user documentation of online transactions
12	Mouse interface
13	Pop-up windows
14	As few screens as possible to accomplish a business function
15	Bilingual support (supports two languages; count as 4 items)
16	Multilingual support (supports more than two languages; count as 6 items)

Rating:

Rating	Description
0	None of the above
1	1-3 of the above
2	4-5 of the above
3	6 or more of the above, but there are no specific user requirements related to efficiency
4	6 or more of the above, and stated requirements for end-user efficiency are strong enough to require design tasks for human factors to be included (for example, minimize key strokes, maximize defaults, use of template)
5	6 or more of the above, and stated requirements for end-user efficiency are strong enough to require use of special tools and processes to demonstrate that the objectives have been achieved

2.2.4 Complex internal processing

Description: Is the business process very complex? Like complicated accounts closing, inventory tracking, heavy tax calculation etc.?

Rating factors: There are 5 factors which govern how this point is rated:

#	Complex internal processing factor
1	Sensitive control (for example, special audit processing) and/or application specific security processing
2	Extensive logical processing
3	Extensive mathematic processing
4	Much exception processing resulting in incomplete transactions that must be processed again. For example, incomplete ATM transactions caused by TP (Teleprocessing) interruption, missing data values or failed edits
5	Complex processing to handle multiple input/output possibilities. For example, multimedia or device independence

Rating:

Rating	Description
0	None of the above
1	1 of the above
2	2 of the above
3	3 of the above
4	4 of the above
5	5 of the above

2.2.5 Code must be reusable

Description: Do we intend to keep the reusability high? If so, the design complexity will increase.

Rating:

Rating	Description
0	No reusable code
1	Development to cater for later enhancement within the application

Rating	Description
2	Less than 10% code to be made reusable for other applications
3	10% to 20% code to be made reusable
4	20% to 35% code to be made reusable Or: The application was specially packaged and/or documented to ease to re-use, and the application is customized by the user at source code level
5	Over 35% code to be made reusable Or: The application was specially packaged and/or documented to ease to re-use, and the application is customized for use by means of user parameter maintenance

2.2.6 Easy to install

Description: Is client looking for installation ease? By default, we get many installers which create packages. But the client might be looking for some custom installation, probably depending on modules. One of our client has a requirement that when the client wants to install, he can choose which modules he can install. Is the requirement such that when there is a new version there should be auto installation? These factors will count when assigning value to this factor.

Rating:

Rating	Description
0	No special considerations were stated by the user, and no special set up is required for installation
1	No special considerations were stated by the user but special set up is required for installation
2	Conversion and installation requirements were stated by the user, and conversion and installation guides were provided and tested. The impact of conversion on the project is not considered to be important
3	Conversion and installation requirements were stated by the user, and conversion and installation guides were provided and tested. The impact of conversion on the project is not considered to be important
4	In addition to 2 above, automated conversion and installation tools were provided and tested
5	In addition to 3 above, automated conversion and installation tools were provided and tested

2.2.7 Easy to use

Description: Is user friendliness a top priority?

Rating:

Rating	Description
0	No special operational considerations other than the normal backup procedures were stated by the user
1-4	1, some, or all of the following items apply to the application. Select all that apply. Each item has a point value of one, except as noted otherwise: Effective start-up, backup, and recovery processes were provided, but operator intervention is required Effective start-up, backup, and recovery processes were provided, but no operator intervention is required (count as 2 items) The application minimizes the need for tape amounts The application minimizes the need for paper handling
5	The application is designed for unattended operation. Unattended operation means no operator intervention is required to operate the system other than to start up or shut down the application. Automatic error recovery is a feature of the application

2.2.8 Portable

Description: Is the customer also looking for cross platform implementation?

Rating:

Rating	Description
0	Application should cater to only one operating system
1	Application should cater to only one type of family of operating system. For example, the application will cater to Windows OS family containing Windows XP, Windows 2000, etc
2	Application should cater to two different families of operating system. For example, Windows and Linux
3	Application should cater to three different families of operating system
4	Application should cater to four different families of operating system
5	Application should cater to five different families of operating system

2.2.9 Easy to change

Description: Is the customer looking for high customization in the future? That also increases the architecture design complexity and hence this factor.

Rating factors: There are 5 factors which govern how this point is rated:

#	Easy to change factor
1	Flexible query and report facility is provided that can handle simple requests. For example, and/or logic applied to only one internal logical file (count as one item)
2	Flexible query and report facility is provided that can handle requests of average complexity. For example, and/or logic applied to more than one internal logical file (count as two items)
3	Flexible query and report facility is provided that can handle complex requests. For example, and/or logic combinations on one or more internal logical files (count as three items)
4	Business control data is kept in tables that are maintained by the user with online interactive processes, but changes take effect only on the next business day
5	Business control data is kept in tables that are maintained by the user with online interactive processes and changes take effect immediately (count as two items)

Rating:

Rating	Description
0	None of the above
1	1 of the above
2	2 of the above
3	3 of the above
4	4 of the above
5	5 of the above

2.2.10 Concurrent

Description: Is the customer looking at large number of users working with locking support? This will increase the architecture complexity and hence this value.

Rating:

Rating	Description
0	No concurrency required

Rating	Description
1	Concurrency required in 10% project data
2	Concurrency required in 30% project data
3	Concurrency required in 50% project data
4	Concurrency required in 70% project data
5	Concurrency required in 100% project data

2.2.11 Including special security features

Description: Is the customer looking at having heavy security like SSL? Or do we have to write custom code logic for encryption?

Rating:

Rating	Description
0	Security aspect is not important
1	Simple third party installation will take care of security
2	For implementing security, third party API's needed to be incorporated in coding. 100% of security is incorporated by using third party API through coding. But the API is well known in the market and easy to understand
3	API is not well known and needs a larger understanding curve to understand
4	Application is combination of third party API's and custom security
5	Full security of application is incorporated by custom coding

2.2.12 Providing direct access for third parties

Description: Does the project depend in using third party controls? For understanding the third-party controls and studying its pros and cons, considerable effort will be required. So, this factor should be rated accordingly.

Rating:

Rating	Description
0	No access needed for third parties
1	Only 5% of functionality needed to be accessed by third party software.
2	Only 10% of functionality needed to be accessed by third party

Rating	Description
	software.
3	Only 20% of functionality needed to be accessed by third party software.
4	Only 50% of functionality needed to be accessed by third party software.
5	100% of functionality needed to be accessed by third party software.

2.2.13 Special user training facilities required

Description: Will the software from user perspective be so complex that separate training has to be provided? So this factor will vary accordingly.

Rating:

Rating	Description
0	No user training required
1	Simple instructions are required to make user understand the system
2	Help files are supplied to user which will be referred by user during using the software
3	With help files user has to be provided with expert guidance in initial stage
4	Special training needed to be provided
5	Special training is required and certification has to be acquired in order that user is eligible to use the product

2.3 Guidelines for environment factors

2.3.1 Familiar with FPT software process

Description: Are all the people working in the project familiar with FPT software development process?

Rating:

Rating	Description
0	None
1	Less than 20% familiarity
2	20% to 40% familiarity
3	40% to 60% familiarity
4	60% to 80% familiarity
5	80% to 100% familiarity

2.3.2 Application experience

Description: How much is the team experience with the type of application being built, or experience with different types of applications?

Rating:

Rating	Description
0	All the team members are novices
1	20% application experience (or a few of the team have 1-1.5 years of experience)
2	40% application experience (or a few of the team have more than 2 years of experience)
3	60% application experience (or most of the team have more than 4 years of experience)
4	80% application experience (or most of the team have more than 6 years of experience)
5	100% application experience (all the team members are experienced)

2.3.3 Paradigm (e.g. Object–oriented) experience

Description: How much is the experience of the development paradigm? For example, with OO paradigm, use-case documents are inputs to object oriented design, it's important that people on the project should have basic knowledge of OOP concepts.

Rating:

Rating	Description
0	No experience
1	Only theoretical experience
2	More than 1 year of experience

Rating	Description
3	More than 2 years of experience
4	More than 4 years of experience
5	More than 8 years of experience

2.3.4 Lead analysis capability

Description: How is the analyst who is leading the project? Does he have enough knowledge of the domain?

Rating:

Rating	Description
0	No lead analyst in the project (or the lead analysis is a novice)
1	1 year of experience lead analyst
2	2 to 3 years of experience lead analyst
3	3 to 4 years of experience lead analyst
4	4 to 6 years of experience lead analyst
5	Above 6 years of experience lead analyst

2.3.5 Motivation

Description: Are the programmers motivated for working on the project? Instability in the project will always lead to people leaving half way through their source code. And the hand over becomes really tough. This factor you can put according to how software industry is going on? Example, if the software market is very good, put this at maximum value. More good the market, more the jobs and more the programmers will jump.

Rating:

Rating	Description
0	No motivation
1	Low motivation. Team works only when being directed. No special initiative from the team members
2	5% of team are high motivated and have self initiative. 90% of team work only when being directed; no special initiative from these team members
3	20% of team are high motivated and have self initiative. 80% of team work only when being directed; no special

Rating	Description
	initiative from these team members
4	50% of team are high motivated and have self initiative. 50% of team work only when being directed; no special initiative from these team members
5	High motivation and self initiation in all members

2.3.6 Stable requirements

Description: Is the client clear of what he wants? Client's expectations are the most important factor in the stability of requirements. If the client is of highly changing nature, put this value to maximum.

Rating:

Rating	Description
0	No satiability. Every meeting with customer changes around 80% deviation from original requirements
1	Requirement changes around 60% of the original requirement
2	Requirement changes around 40% of the original requirement
3	Requirement changes around 20% of the original requirement
4	Requirement changes around 5% of the original requirement
5	Stable

2.3.7 Part time staff

Description: Are there part-time staff in projects, like consultants etc.?

Rating:

Rating	Description
0	No part time staff
1	10% of members are part time staff
2	30% of members are part time staff
3	50% of members are part time staff
4	80% of members are part time staff
5	100% of members are part time staff

2.3.8 Difficult programming language

Description: How much is the language/technology complexity?

Rating:

Rating	Description
0	One week needed to pick up the language/technology (e.g. HTML, JavaScript, etc.)
1	Two weeks needed to pick up the language/technology (e.g. Visual Basic, Visual Basic .NET, etc.)
2	One month needed to pick up the language/technology (e.g. C, C++, C#, Java, etc.)
3	Special training needed for the language/technology (e.g. Ingres/OpenROAD, J2EE, .NET Remoting, SOA, etc.) Or: Special training needed for the well-documented third-party framework/API on which the team develops
4	Special training needed for the language/technology and need help during the project (Any well-known language/technology above but the team must use it against a not well-documented framework/API)
5	Difficult. Need only experience people (e.g. Assembly)

3 REFERENCES

No.	Code/URL	Description
1	Template_EstimateUCP.xls	FSOFT UCP template
2	03e-HD/PM/HDCV/FSOFT	FSOFT software estimation guideline
3	Gustav Karner's research	Resource Estimation for Objector Projects
4		Cockburn, A., *Writing Effective Use Cases*, Addison-Wesley, Boston, MA, 2001
5	Shivprasad Koirala's research	How to prepare software quotation
6	Kirsten Ribu's thesis	Estimating Object-Oriented Software Projects with Use Cases (University of Oslo Department of Informatics)
7	Bente Anda's research	Estimating Software Development Effort based on Use Cases – Experiences from Industry
8	Sofia Nystedt's master thesis and Claes Sandros' bachelor thesis	Software Complexity and Project Performance (Department of Informatics School of Economics and Commercial Law at the university of Gothenburg)
9	Leslee Probasco's research	What About Function Points and Use Cases? (Rational Software Canada)
10	Gautam Banerjee's research	Use Case Points – An estimation Approach

Approver	Reviewer	Creator
Nguyen Lam Phuong	**Dao Duy Cuong**	**Pham Minh Ngoc**