Improving Use Case Point based on Function Point Mechanism

2016. 2. 17.

Bokyung Park

Hongik Univ. Selab

Advisor : R. Young Chul Kim

Table of Contents

1. Research Motivation

- According to advanced software industries, it is increasing software production size system. Therefore, also increases damage with occurring just one defect.
- That is, the bigger software size is, the more an error possibly happens.

It is very important to forecast software estimation, which can adjust man a month, cost, and time of software development for high quality software.

How to estimate software? We adapt UCP mechanism.

- need to identify right requirements for minimizing software error.

- need to calculate UCP complexity, that is, the higher a complexity is, the more an error possibly happens

- For high quality software, Need to be prior(priorize) requirements (use cases) for developing high complexity of use cases first of all.

1. Research Motivation

Previous Research

- Most cases, they used Function Point for SW Project effort estimation

- Extracted and verified the priority of requirements based on UCP

  [So Young Moon, "Verification of Requirements Extraction and Prioritization using Use Case Points"]

  - Problems :
    - ① This method can not make 'effort estimation' results
    - ② difficult to make a systematic system planning
    - ③ the use case weight determined based on the number of transactions in the UCP (still undefined the size of use case)

Our idea

❖ Our idea uses the use case point (UCP) to the effort estimation of SW for an automobile supplies management system

2. Why use UCP, but not FP?

❖ FP Definition

- A function point is a "unit of measurement" to express the amount of business functionality an information system (as a product) provides to a user.
- Function points measure software size.
- The cost (in dollars or hours) of a single unit is calculated from past projects.

❖ UCP Definition

- Use Case Points (UCP) is a software estimation technique used to forecast the software size for software development projects.
- The concept of UCP is based on the requirements for the system being written using use cases, which is part of the UML set of modeling techniques.
- The software size (UCP) is calculated based on elements of the system use cases with factoring to account for technical and environmental considerations.
- The UCP for a project can then be used to calculate the estimated effort for a project.

2. Why use UCP, but not FP?

❖ With FP, They had used the previous system which are developed with the procedural language.

❖ But now, in most of systems, we are developed by object-oriented language.

❖ So, we used UCP for automobile supplies management system to measure SW effort(estimation).

⇒ Which mechanisms are more important to measure SW cost estimation?

3. Prioritization based on Improved UCP

❖ **Use Case Point**

- Developed by Gustav Karner
- Actors and use cases in a use case diagram are used to measure the number of use cases, sizes and complexity.

❖ **The Problem of the previous Use Case Point (UCP)**

① The UCP does not indicate the structure of a specific use case or how to write it.
=>So, vary Use case model and specification per each user(developer)

② The UCP does not allow for "Include" and "Extends" relations between use cases.

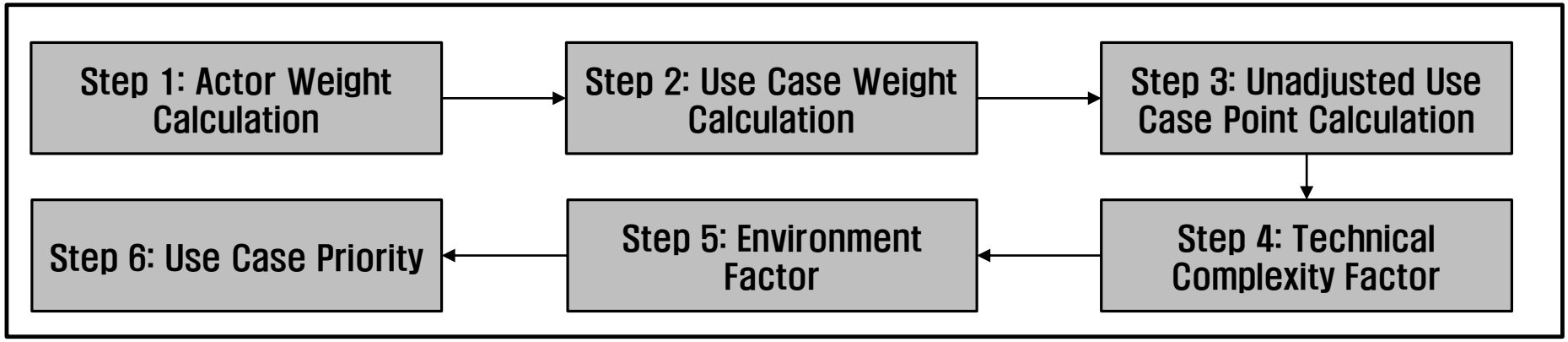
③ UCP weight value was determined by the number of transactions in the UCP. => inaccurately measures use case due to the same size per range.

▪ Our Method

① Subdivide the types and weights of actors and use cases.

② Just add a weight of 0.25 to the use case of "Include," "Extends" relations based on Periyasamy's method

3. Our Improved UCP Process for Use Case Priority



❖ Step 1 : Actor Weight Calculation

- We improve the existing actor's weight values based on Periyasamy's method.
- But still two problems:

① have the seven types of actors.(Very Simple, Simple, Less Average, Average, Complex, Very Complex, Most Complex)

② must separately analyze Primary Actor and Secondary Actor.

∴ So, need more time to calculate the actor's weight value.

=> We improve to decide five actors.

Actor Type	Classification of Actors	Weight
Very Simple	Specialized Actor	0.5
Simple	Actor with 1<number of associations<=3	1
Average	Actor with 3<number of associations<=5	1.5
Complex	Actor with 5<number of associations<=8	2

3. Our Improved UCP Process for Use Case Priority

❖ Step 2 : Use Case Weight Calculation

- A different weight is allocated to each use case.
- Use Case Weight is classified as follows

Use Case Type	Classification of Actors	Weight
Simple	Number of transactions <= 2	0.5
Average	2 < Number of transactions <= 4	1
Complex	4 < Number of transactions <= 6	2
Very Complex	Number of transactions > 6	3

- A weight of 0.25 is added to a use case that involves any "Include" or "Extend" relation.
- Each extracted use case is prioritized based on its weight.

❖ Step 3 : Unadjusted Use Case Point(UUCP) Calculation

- UUCP = Actor Weight + Use Case Weight

3. Our Improved UCP Process for Use Case Priority

❖ **Step 4 : Technical Complexity Factor(TCF) Calculation**

- A Weight between 0(no effect) and 5(large effects) is applied to each component.

Technical Factor	Factor Description	Weight
T1	Distributed System	2
T2	Response or Throughput Objectives	1
T3	End-User Efficiency	1
T4	Complex Internal Processing	1
T5	Code must be reusable	1
T6	Easy to install	1
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 to third-party SW	1
T13	Special user Training facility is required	1

❖ **Step 5 : Environment Factor(EF) Calculation**

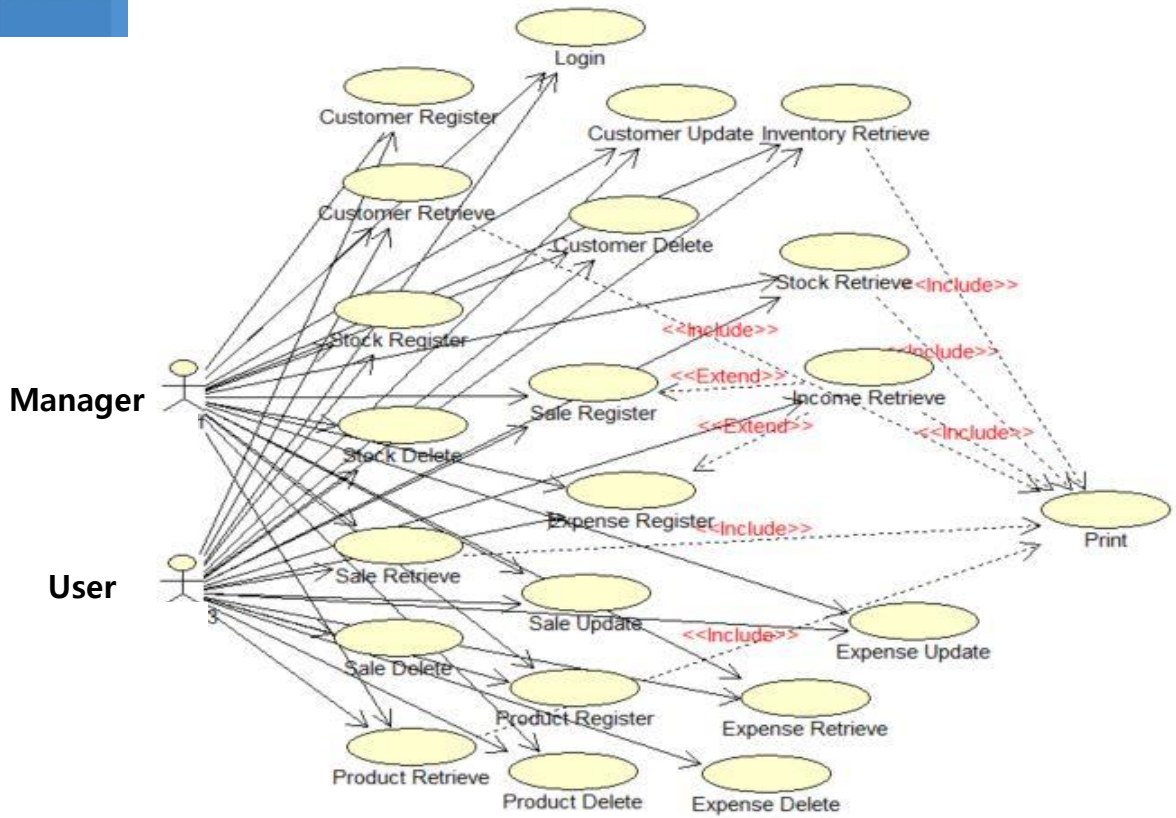
- The EF is calculated by applying a weight between 0 and 5.

Environment Factor	Factor Description	Weight
E1	Familiarity with UML	1.5
E2	Part-Time Workers	-1
E3	Analyst Capability	0.5
E4	Application Experience	0.5
E5	Object Oriented Experience	1
E6	Motivation	1
E7	Difficult Programming Language	-1
E8	Stable Requirements	2

❖ **Step 6 : The Priority based on UCP**

- UCP = UUCP * TCF * EF
- Priority: Determined based on the extracted UCP values.

4. Case Study
– Use Case Diagram for an automobile supplies management system



- **Results**
  - The number of Use Cases:22
  - The number of Actors: 2
  - Include: 6, Extend: 2

4.1 How to calculate UCP?

Calculated UAW and UUCW

UUCP = UAW + UUCW=2

No	Use Case	Unadjusted Actor Weight(UAW) (Manager) Actor Weight	(User) Actor Weight	Actor Weight	UAW	Unadjusted UseCase Weight(UUCW) Basic Flow	Alternative Flow	Exceptional Flow	Include/ Extends	Total Transaction	Use Case Weight	UUCP
UC1	Login	1									1	2
UC2	Customer Register	1									0.5	1.5
UC3	Customer Update	1	1	2	1	1	0	0	0	1	0.5	1.5
UC4	Customer Retrieve	1	1	2	1	1	1	0	0.25	2.25	1	2
UC5	Customer Delete	1	1	2	1	1	0	0	0	1	0.5	1.5
UC6	Stock Register	3	1	4	1.5	1						
UC10	Sale Retrieve	3	1	2	1	1	1	0	0.25	2.25	1	2
UC11	Sale Update	3	1	2	1	1	1	0	0	2	0.5	1.5
UC12	Sale Delete	0.5	0.5	1	0.5	1	1	0	0	2	0.5	1
UC13	Product Register	3	1	4	1.5	1	0	0	0	1	0.5	2
UC14	Product Retrieve	3	1	4	1.5	1	1	0	0.25	2.25	1	2.5
UC15	Product Delete	3	1	4	1.5	1	0	0	0	1	0.5	2
UC16	Inventory Retrieve	1.5	1.5	3	1	1	1	0	0.25	2.25	1	2
UC17	Income Retrieve	1.5	1.5	3	1	1	3	0	0.25	4.25	2	3
UC18	Expense Register	1.5	1	2.5	1	1	0	1	0	2	0.5	1.5
UC19	Expense Update	1	1	2	1	1	0	0	0	1	0.5	1.5
UC20	Expense Retrieve	1	1	2	1	1	3	0	0.25	4.25	2	3
UC21	Expense Delete	0.5	0.5	1	0.5	1	0	0	0	1	0.5	1
UC22	Print	0	0	0	0	1	1	0	1.5	3.5	2	2

UC4, if exists an “Include” relationship => add 0.25

Actor with 1<Number of Associations<=3,
UC4(Customer Retrieve) Actor Weight=2 => UAW: Simple(1)

UC4 Total Transaction: 2.25 ,
2<Number of transactions <= 4: UUCW:Simple(1)
=> Simple(1)

Actor Weight

Use Case Weight

4.2 How to calculate UCP, Priority?

Calculated UCP, Priority and Total Estimate

		TCF1	TCF2	TCF3	TCF4	TCF5	TCF7	TCF9	TCF11	TCF13	TCF VALUE	UCP	PRIORITY
		0 ~ 5											
No	Use Case	2	1	1	1	1	0.5	1	1	1			
UC1	Login	0	2	1.5	0	0	1.5	0	2	1	8	16	12
UC2	Customer Register	0	2	1.5	1	2	2	0	0	0	8.5	12.75	14
UC3	Customer Update	0	1	2	1	2	2	3	0	0	11	16.5	11
UC4	Customer Retrieve	0	1	2	0	0	2	1	0	0	6	12	15
UC5											2.8	4.2	22
UC6											3	4.5	21
UC7	Stock Retrieve	0	3	3	4	1	4	1	2	0	18	36	1
UC8											12.5	18.75	8
UC9											4	6	20
UC10											5.5	11	16
UC11	Sale Update	0	3	3	4	1	4	1	2	1	19	28.5	4
UC12											7.5	17.5	10
UC13											12		9
UC14	Product Retrieve	0	2	2	1	0	2	1	2	1	13	26	5
UC15											13	19.5	7
UC16											12	30	3
UC17	Income Retrieve	0	1	2	0	0	0	0	0	0	3	10.5	18
UC18	Expense Register	0	1	2.3	0	0	1	2	0	0	6.3	9.45	19
UC19	Expense Update	0	2	3	3	1	4	1	0	0	14	21	6
UC20	Expense Retrieve	0	1	3	1	1	3	2	0	0	11	33	2
UC21	Expense Delete	0	2	3	0	1	2	3	0	0	11	11	16
UC22	Print	0	1	2	0	1	1.5	1	0	0	6.5	13	13

UC3 TCF Value= 2*0+ 1*1+ 1*2+ 1*1+ 1*2+ 0.5*2+ 1*3+ 1*0+ 1*0
= 2+ 1+ 2+ 1+ 2+ 1+ 3+ 0+ 0=11

- In this paper, not consider EF value due to the same value 3 of EF per each use case,.

UC3 UCP = UUCP*TCF=1.5*11=16.5

- Determine the priority of use cases based on compassion of the extracted UCP values.

4.3 Prioritization based on existing UCP VS. Improved UCP

The comparison results

No	Use Case	The Priority based on UCP	The Priority based on Improved UCP
UC01	Login	14	12
UC02	Customer Register	16	14
UC03	Customer Update	8	11
UC04	Customer Retrieve	13	15
UC05	Customer Delete	19	22
UC06	Stock Register	15	21
UC07	Stock Retrieve	7	1
UC08	Stock Delete	18	8
UC09	Sale Register	15	20
UC10	Sale Retrieve	4	16
UC11	Sale Update	9	4
UC12	Sale Delete	12	10
UC13	Product Register	16	9
UC14	Product Retrieve	6	5
UC15	Product Delete	17	7
UC16	Inventory Retrieve	4	3
UC17	Income Retrieve	2	18
UC18	Expense Register	10	19
UC19	Expense Update	3	6
UC20	Expense Retrieve	1	2
UC21	Rexpense Delete	11	16
UC22	Print	20	13

I. As comparing with two ways, we got over 10 value, that is, the difference between two approaches on six use cases such as Stock Delete, Sale Retrieve, Product Register, Product Delete, Income Retrieve, Expense Register

- Previous UCP: define Actor(1~3), Use Case(5,10,15)
- Improved UCP: redefine Actor(0.5~2.5), Use Case(0.5~3)

⇒ As with different weight values, we got the different UCP values.

II. About Actor Weight and Use Case Weight,

- Actor weight values are measured with the directive number of the association between an use case and an actor based on "improved UCP mechanism".

⇒ The priority of "Improved UCP" have differently measured with one of "previous UCP".

III. Our Improvement:

- We consider "include" and "extend" relationships on use case diagram to extract UCP value.
- More in detail, classify each weight value of actor and use case.

⇒ Expect possibly to extract the correct UCP value with our improved UCP mechanism.

5. The Contribution of this paper

- With Priority Extraction
  - We got the prioritization of all use case, which means we decides to develop system based on the priority of use cases

- With UCP Extracted,
  - We can recognize which use cases are more complex.

- ~~With Estimation Extracted,~~
  - ~~We can estimated cost of development.~~

6. Conclusion & Future Works

✓ Conclusion

- Improve the priority method of the previous UCP mechanism.
- We extract the priority of use case with our "improved UCP mechanism".
- The Improved Use Case Point is as follows:
  - The improved UCP method measures the actor weights based on the number of direct associations between actors and use cases.
  - The use case weights are measured based on the number of transactions.
  - The number of transactions is adjusted, or lowered for the use case weight compared to the existing UCP.

✓ Future Works

- **We will apply the automobile supplies management system with our proposed method in future studies.**
- **Also Need to validate which approaches are more correct.**
- **We should compare to mechanism: FP and UCP with the real development system.**

Q&A

Thank You