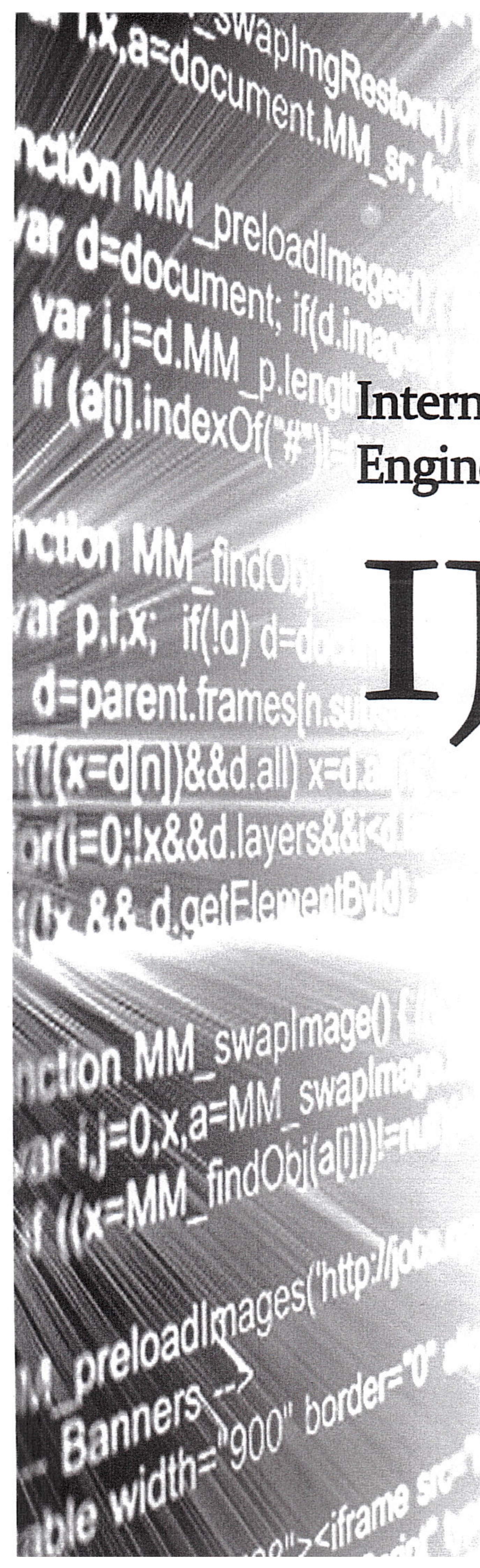
ISSN: 1738-9984


i,x,a=document.MM_sr; for
_swapImgRestore()
nction MM_preloadImages()
var d=document; if(d.images)
var i,j=d.MM_p.length
if (a[i].indexOf("#")!=
nction MM_findObj
var p,i,x; if(!d) d=do
d=parent.frames[n.su
if(!(x=d[n])&&d.all) x=d.a
for(i=0;!x&&d.layers&&i<d
(!x && d.getElementById
nction MM_swapImage()
var i,j=0,x,a=MM_swapImag
if ((x=MM_findObj(a[i]))!=nu
MM_preloadImages("http://job
Banners -->
able width="900" border="0"
<iframe src=

International Journal of Software Engineering and Its Applications

IJSEIA

Vol.7, No.4, July, 2013

SCIENCE & ENGINEERING
RESEARCH SUPPORT SOCIETY

Verification of Requirements Extraction and Prioritization Based on Goal Oriented Use Case Approach by using Use Case Points

So Young Moon* and Robert Young Chul Kim*

**Department of Computer and Information Communication,*
Hongik University, Sejong Campus, 339-701, Korea

{msy, bob}@selab.hongik.ac.kr

Abstract

To collect and analyze the requirements to make the products that customers want is an important starting point in the requirements engineering stage. In the requirement engineering is focus on building a qualitative system through identification and extraction of value requirements. A lot of researches for complete requirements, but they are difficult to determine correct extraction requirements. A purpose of this paper is that a goal approach is graft on to use case approach to reach a target system through goal-oriented requirements. For this, we suggest that the extraction and prioritization of requirements for fitting on the goal approach, and then to make a change an existing value based approach to show the prioritization of requirements based on goal-oriented requirements. It helps to make a target product customers want from extracting a highly importance of use cases and requirements. Beside it gets a benefit of the prioritization of test cases by prioritization of all requirements, and it is possible to measure coverage of test cases quantitatively. Finally, we verify the prioritization of requirements based on goal-oriented use case approach by calculating prioritization about use cases using the Use Case points proposed by Karner [5].

Keywords: *Use Case Point, Function Point, Goal-Oriented Requirements Process, Requirements Prioritization*

1. Introduction

Customers have a meeting with developers about requirements to make a product they want; they anticipate software developed as their demands. However inexact requirements or incorrect requirements cause a redevelopment problem frequently. The cost of error correction increases exponentially depending on the error detection time [1]. For example, if error is detected at the time of requirement collection, the cost of error correction would be 3, but at the time of design, the cost would be 5, at the time of coding, 9, at the time of testing, 17, and at the time of production, 160. Therefore, there would be more cost of error correction if detected at the time of latter project phase compared to the one immediately corrected after occurrence of error. As a result, requirements collection and analysis in the software development life-cycle is an important step for successful software development [9].

For building a system as customer's demands, many researches for complete requirements about collecting and analyzing requirements, but they are difficult to determine correct extraction requirements [2, 3].

Cockburn classified a goal as a core element of use case in Goals and Use Case (1997), and introduced a communication model based theory [7, 8]. And in our previous research, we proposed the extraction of goal-oriented requirements and the method of priority determination by using his goal oriented use case. It is difficult to determine that which requirement has the highest priority, but using the Use Case is helpful to determine priority of requirements [5].

The purpose of this paper is to reach to the goal system through a goal oriented requirements [10] by applying the goal oriented concepts to the use case approach. The reason of prioritizing of requirements is to get a priority of test case by prioritizing all requirements. If test cases are prioritized, it can help to measure test coverage quantitatively. Using this method, however, can determine priority but there is no method of verification. Therefore, in this paper, we verify the priority technique based on the goal-based Use Case using the Use Case Points proposed by Karner.

This paper is organized as follows: In Chapter 2, related work is described such as Use Case Points proposed by Karner. In chapter 3, describing prioritization of requirements by the method of goal oriented use case. In Chapter 4, calculating Use Case Priority using the Use Case Points is discussed. In Chapter 5, the goal-based Use Case technique is verified by the Use Case Points. Finally in Chapter 5, conclusion and future work are discussed.

2. Related Work

UCP (Use Case Points) is developed based on basic concepts of function point by Gustav Karner [5, 9]. It estimates the number of use case, a size of use case, a complexity of use case quantitatively by an actor and a use case in use case diagram for estimation of software size. Use Case Points consider not only a use case complexity itself but also actor which interacts with Use Case [4]. A method of calculation of UCP follows like this. First of all, unadjusted use case point is calculated through an actor and use case described in use case diagrams. Next, the UCP is calculated by using technical complexity factor and environmental factor with unadjusted use case. A technical complexity factor is calculated by selecting parameters that may affect the system. And then an environmental factor is calculated by selecting parameters that may affect efficiency of a project development. The all steps of calculation of UCP as follows.

Step 1, weights are calculated according to an actor's complexity as Figure 1. A complexity of other systems defined by API is simple and a weight is 1, and then other system which interacts via a protocol like TCP/IP is average or a user who interacts via a simple user interface is average, the weight is 2. Next, the complexity of an interactive actor through GUI is complex, and the weight is 3. So Calculation of actor' s weights is UAW (Unadjusted Actor Weights) = ∑ (No of Actors * Weight).

Step 2, weights are calculated according to a complexity of a use case as Figure 1. The weight is decided according to the number of transaction of use case, and if the number of transaction is less than 3, complexity is simple. If the number of transaction is between 4 and 7, the complexity is average, weight is 10. If the number of transaction is more than 8, the complexity is complex, weight is 15. So Calculation of use case' s weights is UUCW (Unadjusted Use Case Weights) = ∑ (No of Use Cases * Weights).

Step 3, UUCW (Unadjusted Use Case Weights) added to UAW (Unadjusted Actor Weights) is UUCP (Unadjusted Use Case Points). A result of step 3 is unadjusted use case, so UCP is calculated by applying weights of adjusted parameters such as technical complexity factors and environmental factors.

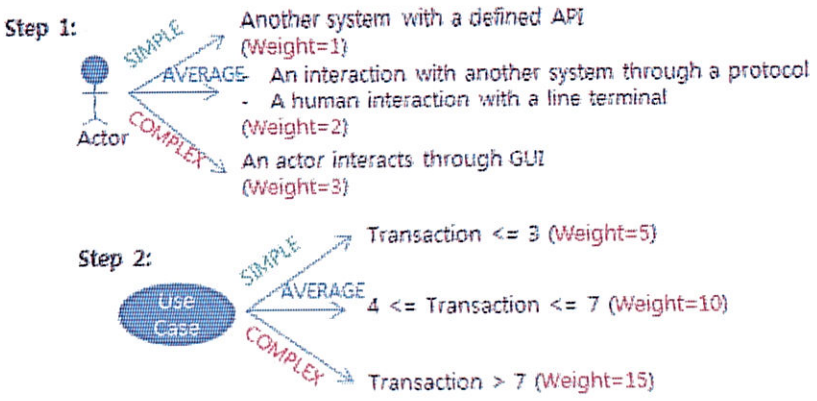

Step 1:
Actor
SIMPLE
AVERAGE
COMPLEX
Another system with a defined API
(Weight=1)
An interaction with another system through a protocol
- A human interaction with a line terminal
(Weight=2)
An actor interacts through GUI
(Weight=3)
Step 2:
Use Case
SIMPLE
AVERAGE
COMPLEX
Transaction <= 3 (Weight=5)
4 <= Transaction <= 7 (Weight=10)
Transaction > 7 (Weight=15)

Figure 1. The Step 1 and Step 2 in the Calculation of UCP

Step 4, 13 TCF (Technical Complexity Factor) and 8 EF (Environmental Factor) is consist of between 0 and 5 levels. So TCF is (0.01 * ∑Fi * Weight) added to 0.6, EF is (-0.03 * ∑Fi * Weight) added to 1.4. In case, Fi is a value of levels. The calculation is as follows.

- TCF = 0.6 + (0.01 * ∑Fi * Weight)
- EF = 1.4 + (-0.03 * ∑Fi * Weight)

Step 5, UCP is calculated by multiplying a result of step 3 by a result of step 4. (UCP = UUCP * TCF * EF).

3. Prioritization of Use Case by using a Goal Oriented Use Case Approach

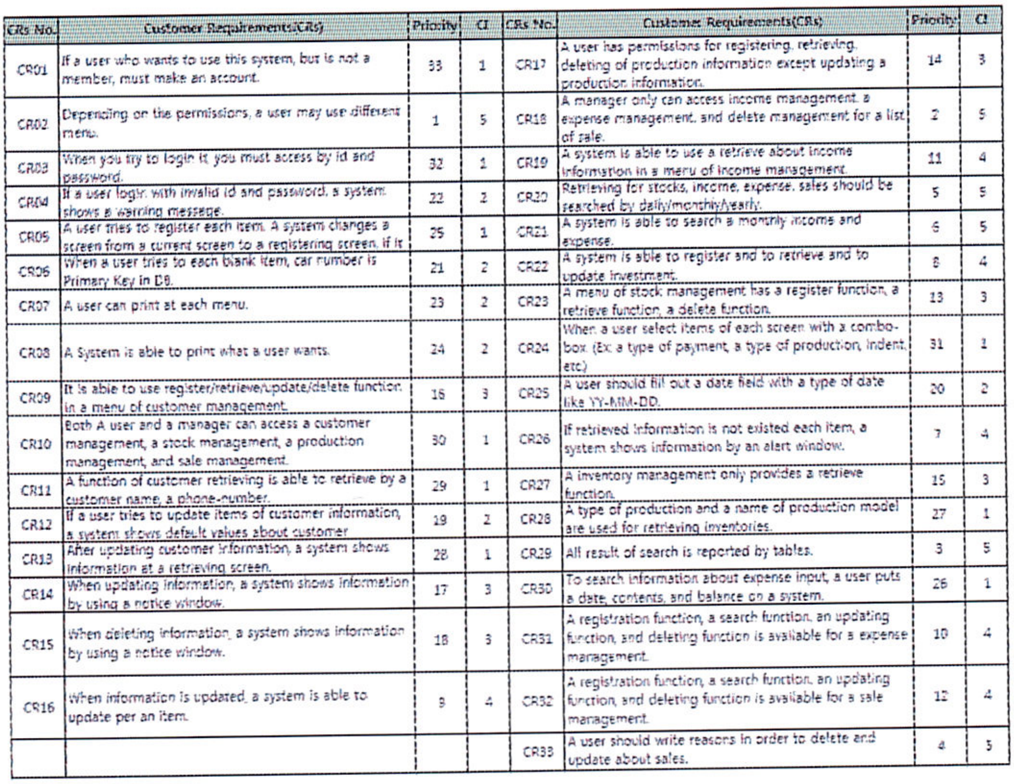
The extraction process of goal-oriented requirements and use case priority can be divided by two parts [6-8]. To extract requirements in developer's point of view is in a step of extraction of goal-oriented requirements. Based on this information, create a list of the requirements contained within each use case must complete the requirements of customers.

TST Sejong multi-shop management program as the case study of this paper is extracted 22 use cases and 33 requirements. Extracted Customer requirements are summarized in Table 1.

Goal-oriented requirements and applying VIRE (Value-Innovative Requirements Engineering) method in the step of determination of use case priority. For determining priority, a customer and a developer determine which requirements are important within extracted customer requirements in a previous step (Extraction of goal-oriented requirements). CI (Customer Importance) has a value of between 1 and 5 according to a priority of requirements.

As shown in Table 1, depending on the ranking of customer requirements, customer importance will be different that means higher ranking value will be high. The customer requirement of CR18 is "A manager only can access income management.", and the rank of requirement is 2 and the customer importance is 5. And the customer requirement of CR08 is "A system is able to print what a user wants.", and it is calculated as the rank of requirement is 24 and customer importance is 2.

Table 1. Priority and CI of Customer Requirements



CRs No.	Customer Requirements(CRs)	Priority	CI	CRs No.	Customer Requirements(CRs)	Priority	CI
CR01	If a user who wants to use this system, but is not a member, must make an account.	33	1	CR17	A user has permissions for registering, retrieving, deleting of production information except updating a production information.	14	3
CR02	Depending on the permissions, a user may use different menu.	1	5	CR18	A manager only can access income management, a expense management, and delete management for a list of sale.	2	5
CR03	When you try to login it you must access by id and password.	32	1	CR19	A system is able to use a retrieve about income information in a menu of income management.	11	4
CR04	If a user login with invalid id and password, a system shows a warning message.	22	2	CR20	Retrieving for stocks, income, expense, sales should be searched by daily/monthly/yearly.	5	5
CR05	A user tries to register each item. A system changes a screen from a current screen to a registering screen. If it	25	1	CR21	A system is able to search a monthly income and expense.	6	5
CR06	When a user tries to each blank item, car number is Primary Key in DB.	21	2	CR22	A system is able to register and to retrieve and to update investment.	8	4
CR07	A user can print at each menu.	23	2	CR23	A menu of stock management has a register function, a retrieve function, a delete function.	13	3
CR08	A System is able to print what a user wants.	24	2	CR24	When a user select items of each screen with a combo-box. (Ex: a type of payment, a type of production, Indent, etc)	31	1
CR09	It is able to use register/retrieve/update/delete function in a menu of customer management.	16	3	CR25	A user should fill out a date field with a type of date like YY-MM-DD.	20	2
CR10	Both A user and a manager can access a customer management, a stock management, a production management, and sale management.	30	1	CR26	If retrieved information is not existed each item, a system shows information by an alert window.	7	4
CR11	A function of customer retrieving is able to retrieve by a customer name, a phone-number.	29	1	CR27	A inventory management only provides a retrieve function.	15	3
CR12	If a user tries to update items of customer information, a system shows default values about customer	19	2	CR28	A type of production and a name of production model are used for retrieving inventories.	27	1
CR13	After updating customer information, a system shows information at a retrieving screen.	28	1	CR29	All result of search is reported by tables.	3	5
CR14	When updating information, a system shows information by using a notice window.	17	3	CR30	To search information about expense input, a user puts a date, contents, and balance on a system.	26	1
CR15	When deleting information, a system shows information by using a notice window.	18	3	CR31	A registration function, a search function, an updating function, and deleting function is available for a expense management.	10	4
CR16	When information is updated, a system is able to update per an item.	9	4	CR32	A registration function, a search function, an updating function, and deleting function is available for a sale management.	12	4
				CR33	A user should write reasons in order to delete and update about sales.	4	5

In Next step, it analyzes the link from matching up between each use case and each requirement. It figures out whether any requirements associated with any use case, will be measured by ranking. It gives 9 point for a strong relation, 3 point for a middle relation, and 1 point for weak relation.

Applying SE (System Elements) on the horizontal axis is when analyzing a matrix in VIRE (Value-Innovative Requirements Engineering) method, that time SE will be extracted by element of software and hardware. However it is difficult to apply SE, because considering only software elements in this paper. Therefore to use each use case unit instead of SE for applying the determination matrix to goal-oriented use case approach as shown in Figure 2.

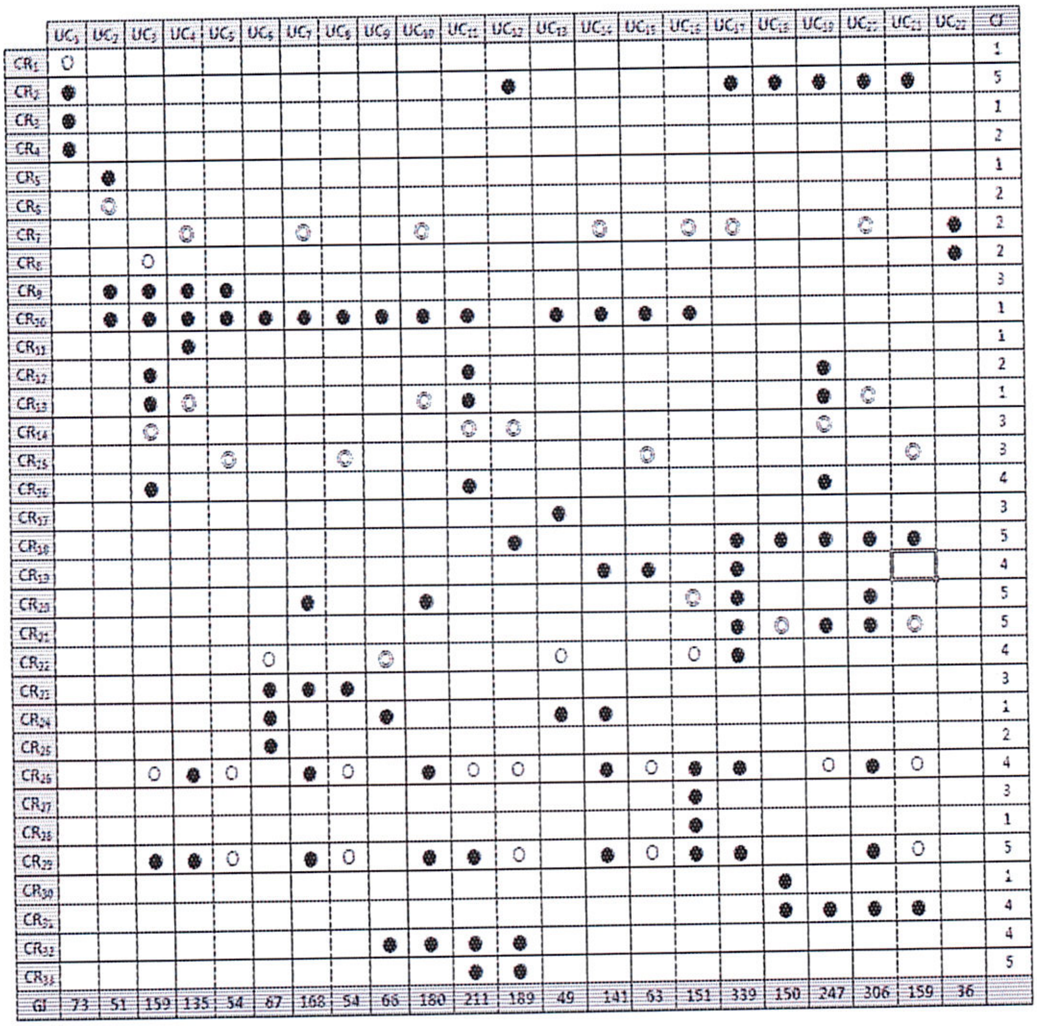
Goal importance is calculated in analyzing the matrix with VIRE method. Grade (R i, j) which is association between the use cases and requirements, multiplies and sum CI (Customer Importance) for deciding goal importance. The formula of Goal importance (GI: Goal Importance) is as follows.

$$GI = \{(CI_i \times R_{i,j}) + (CI_2 \times R_{i,j}) + \cdot \cdot \cdot + (CI_i \times R_{i,j})\}$$
$$= \sum_{i,j=1}^{n} (CI_i \times R_{i,j})$$

In formula (1), (R i, j) represents an relation grade by analyzing relation. We can see that UC1 is related to CR1, CR2, CR3, and CR4 in Figure 2. GI of use case is calculated by formula (1).

UC1's GI = ((CR1xR1,1)+(CR2xR1,2)+(CR3xR1,3)+(CR4xR1,4)) =
(1x1)+(5x9)+(1x9)+(2x9) = 1+45+9+18 = 73

UC2's GI = ((CR5xR2,5)+(CR6xR2,6)+(CR9xR2,9)+(CR10xR2,10)) =
(1x9)+(2x3)+(3x9)+(1x9) = 9+6+27+9 = 51



	UC_1	UC_2	UC_3	UC_4	UC_5	UC_6	UC_7	UC_8	UC_9	UC_{10}	UC_{11}	UC_{12}	UC_{13}	UC_{14}	UC_{15}	UC_{16}	UC_{17}	UC_{18}	UC_{19}	UC_{20}	UC_{21}	UC_{22}	CI
CR_1	○																						1
CR_2	●											●					●	●	●	●	●		5
CR_3	●																						1
CR_4	●																						2
CR_5		●																					1
CR_6		◎																					2
CR_7				◎			◎			◎				◎		◎	◎			◎		●	2
CR_8			○																			●	2
CR_9		●	●	●	●																		3
CR_{10}		●	●	●	●	●	●	●	●	●	●		●	●	●	●							1
CR_{11}				●																			1
CR_{12}			●								●								●				2
CR_{13}			●	◎						◎	●								●	◎			1
CR_{14}			◎								◎	◎							◎				3
CR_{15}					◎			◎							◎						◎		3
CR_{16}			●								●								●				4
CR_{17}													●										3
CR_{18}												●					●	●	●	●	●		5
CR_{19}														●	●		●						4
CR_{20}							●			●						◎	●			●			5
CR_{21}																	●	◎	●	●	◎		5
CR_{22}						○			◎				○			○	●						4
CR_{23}						●	●	●															3
CR_{24}						●			●				●	●									1
CR_{25}						●																	2
CR_{26}			○	●	○		●	○		●	○	○		●	○	●	●		○	●	○		4
CR_{27}																●							3
CR_{28}																●							1
CR_{29}			●	●	○		●	○		●	●	○		●	○	●	●			●	○		5
CR_{30}																		●					1
CR_{31}																		●	●	●	●		4
CR_{32}									●	●	●	●											4
CR_{33}											●	●											5
GI	73	51	159	135	54	67	168	54	66	180	211	189	49	141	63	151	339	150	247	306	159	36	

Figure 2. Decision Matrix for Requirements

UC3's GI = ((CR8xR3,8) + (CR9xR3,9) + (CR10xR3,10) + (CR12xR3,12) + (CR13xR3,13) + (CR14xR3,14) + (CR16xR3,16) + (CR26xR3,26) + (CR29xR3,29)) = (2x1)+(3x9)+(1x9)+(2x9)+(1x9)+(3x3)+(4x9)+(4x1)+(5x9)=2+27+9+18+9+9+36+4+45=159

UC4's GI = ((CR7xR4,7) + (CR9xR4,9) + (CR10xR4,10) + (CR11xR4,11) + (CR13xR4,13) + (CR26xR4,26) + (CR29xR4,29))=
(2x3)+(3x9)+(1x9)+(1x9)+(1x3)+(4x9)+(5x9) = 6+27+9+9+3+36+45 = 135

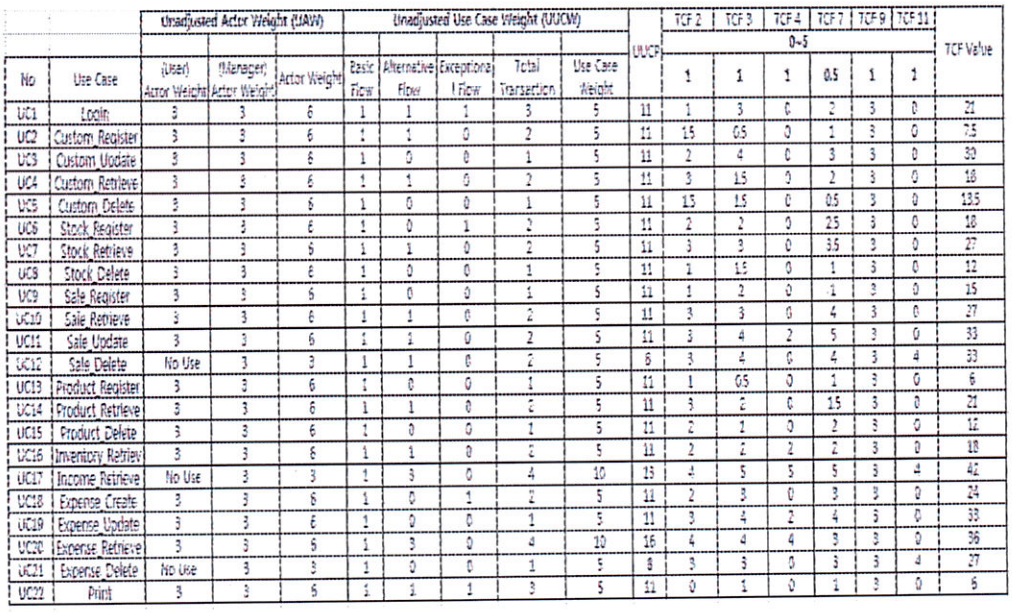
As follows, the goal priority of use case is decided by measuring the goal importance through analyzing the matrix. The priority of use case according to the goal-oriented use case approach is shown as follows:

Income_Retrieve (UC17) →Expense Retrieve (UC20) →Expense_Update (UC19) →
Sale_Update (UC11) →Sale_Delete (UC12) →Sale_Retrieve (UC10) →
Stock_Retrieve (UC7) → Customer_Update (UC3) → Expense_Delete (UC21) →
Inventory_Retrieve (UC16) → Expense_ Register (UC18) → Product_Retrieve (UC14)→
Customer_Retrieve (UC4) → Login (UC1) → Stock_ Register (UC6) →
Sale_ Register (UC9) → Product_Delete (UC15) → Customer_Delete (UC5) →
Stock_Delete (UC8) → Customer_Register (UC2) → Product_ Register (UC13) →
Print (UC22)

4. Prioritization of Use Case by using Use Case Points

After extracting use case depending on requirements of Table 1, Use Case Points is calculated. We applied the actor weight, Use Case weight, weights of the technical complexity factor and environmental factor as proposed in the Use Case point calculation by Karner.

Table 2. Calculated UCP and TCF [9]



		Unadjusted Actor Weight (UAW)			Unadjusted Use Case Weight (UUCW)					UUCP	TCF 2	TCF 3	TCF 4	TCF 7	TCF 9	TCF 11	TCF Value
											0~5						
No	Use Case	(User) Actor Weight	(Manager) Actor Weight	Actor Weight	Basic Flow	Alternative Flow	Exceptional Flow	Total Transaction	Use Case Weight		1	1	1	0.5	1	1	
UC1	Login	3	3	6	1	1	1	3	5	11	1	3	0	2	3	0	21
UC2	Custom_Register	3	3	6	1	1	0	2	5	11	1.5	0.5	0	1	3	0	7.5
UC3	Custom_Update	3	3	6	1	0	0	1	5	11	2	4	0	3	3	0	30
UC4	Custom_Retrieve	3	3	6	1	1	0	2	5	11	3	1.5	0	2	3	0	18
UC5	Custom_Delete	3	3	6	1	0	0	1	5	11	1.5	1.5	0	0.5	3	0	13.5
UC6	Stock_Register	3	3	6	1	0	1	2	5	11	2	2	0	2.5	3	0	18
UC7	Stock_Retrieve	3	3	6	1	1	0	2	5	11	3	3	0	3.5	3	0	27
UC8	Stock_Delete	3	3	6	1	0	0	1	5	11	1	1.5	0	1	3	0	12
UC9	Sale_Register	3	3	6	1	0	0	1	5	11	1	2	0	1	3	0	15
UC10	Sale_Retrieve	3	3	6	1	1	0	2	5	11	3	3	0	4	3	0	27
UC11	Sale_Update	3	3	6	1	1	0	2	5	11	3	4	2	5	3	0	33
UC12	Sale_Delete	No Use	3	3	1	1	0	2	5	6	3	4	0	4	3	4	33
UC13	Product_Register	3	3	6	1	0	0	1	5	11	1	0.5	0	1	3	0	6
UC14	Product_Retrieve	3	3	6	1	1	0	2	5	11	3	2	0	1.5	3	0	21
UC15	Product_Delete	3	3	6	1	0	0	1	5	11	2	1	0	2	3	0	12
UC16	Inventory_Retriev	3	3	6	1	1	0	2	5	11	2	2	2	2	3	0	18
UC17	Income_Retrieve	No Use	3	3	1	3	0	4	10	13	4	5	5	5	3	4	42
UC18	Expense_Create	3	3	6	1	0	1	2	5	11	2	3	0	3	3	0	24
UC19	Expense_Update	3	3	6	1	0	0	1	5	11	3	4	2	4	3	0	33
UC20	Expense_Retrieve	3	3	6	1	3	0	4	10	16	4	4	4	3	3	0	36
UC21	Expense_Delete	No Use	3	3	1	0	0	1	5	8	3	3	0	3	3	4	27
UC22	Print	3	3	6	1	1	1	3	5	11	0	1	0	1	3	0	6

In this chapter, we explain the calculating method of priority of requirements of TST Sejong multi-shop management program by using the calculation methods described in Step 1 to Step 6. Table 2 shows the results of priority of requirements of TST Sejong multi-shop management program by using the Use Case Points [9].

Information in Step 1 to Step 3, refer to Table 2 of the Use Case Points Calculation and check each step-by-step calculations.

- Step 1: Actor Weight Calculation

Actor can be categorized into two: user and supervisor. How to calculate an actor weight in the Use Case points is to calculate an actor as simple, average and complex actor. However, in this paper, an actor's weight is calculated in each Use Case for priority of requirements.

In case of the income query (UC17) which can be accessed only by a supervisor, the weight of a general user is none, and there is actor weight only for the supervisor. Since an actor and system interact with each other through the GUI, the weight of an actor on the income query is 3 which is complex. In case of the customer register (UC2), since a general user and supervisor can access and interact through the GUI, the weight for a general user is 3, and for a supervisor is 3; total weight of the actor is 6.

- Step 2: Use Case Weight Calculation

The Use Case weight is simple if the number of transactions is less than 3, average if the number of transactions is between 4 and 7, and complex if the number of transactions is over 8. Total sum of all calculate the Use Cases weights becomes the final Use Case weight. However, in this paper, total sum is not calculated but each Use Case weight is calculated for calculating priority of the Use Cases.

In Table 2, the Use Case weight is assigned 10 since basic flow 1, alternative flow 3, and exception flow 0 which makes the total transaction 4. Finally unadjusted Use Case point is calculated as 13 by summing the actor weight and Use Case weight calculated in step 1.

- Step 3: Unadjusted Use Case Point Calculation

In Table 2, an unadjusted Use Case point is calculated by summing actor weight and Use Case weight by Use Cases. UC2 customer register is 11 by calculating (Actor Weight: 6) + (Use Case Weight: 5), and UC17 income query is 13 by (Actor Weight: 3) + (Use Case Weight: 10). Table 1 shows all the calculated results which sum all the weights of actor and Use Case in all 22 Use Cases.

- Step 4: Technical Complexity Factor Calculation

To calculate the technical complexity factor in the Use Case points, the weights are given between 0 (no effect) and 5 (great effect) to each factor in terms of overall system effect. In this paper, weights are given by 0.5 units for detailed priority of requirements.

As shown Table 2, calculating technical complexity factors depending on use case, and then multiplies it to UUCP (Unadjusted Use Case Points).

- TCF 2 : Application performance objectives, in either response or throughput
- TCF 3 : End user efficiency (on-line)
- TCF 4 : Complex internal processing
- TCF 7 : Operational ease, usability
- TCF 9 : Changeability
- TCF 11 : Special security features

In this case of Income_Retrieve (UC17) in Table 2, TCF 2 is 4 points, TCF 3 is 5 points, TCF 4 is 5 points, TCF 7 is 5 points, TCF 9 is 3 points, and TCF 11 is 4 points. Multiply these values and the weights of each, it takes the highest priority as TCF 2 (4 points) * 1 +

TCF 3 (5 points) + TCF 4 (5 points) * 1 + TCF 7 (5 points) + TCF 9 (3 points) + TCF 11 (4 points) * 1 = 23.5.

- Step 5: Environmental Factor Calculation

For the environmental factor, it is calculated by applying weights between 0 to 5depending on its category such as familiarity of the life-cycle model during the project (1.5), experience on the area (0.5), experience on development methodology used (1), ability of analyzer (0.5), motivation of the team (1), stabilization of the requirements (2), part-time team member use (-1) and difficult programming language use (-1). In this paper, weight for the environmental factor is given 3 to all the Use Cases and the calculated value is 13.5. However, this value is discarded since all have the same value.

- Step 6: Priority based on the Use Case Points

After completing all the calculations between Step 1 to Step 5, priority of UCP can be determined by calculating overall priority.

5. Verification of Goal-oriented Requirements Priority using Use Case Priority with Use Case Point

In this chapter, goal-oriented requirements priority and Use Case Point priority are compared and evaluated. One result is from the priority technique using previously proposed goal-oriented requirements, and the other result is from the Use Case priority using the Use Case Points. Through customer's requirements in Table 1, Use Case priority is calculated using the Use Case Points.

In Figure 3, priority of login is 15 in UCP, 14 in GORP. Priority of Custom_Register is 20 in UCP and GORP. Priority of Custom_Update is 8 in UCP and GORP. Priority of Custom_Retreive is 13 in UCP and GORP. Priority of Custom_Delete is 18 in UCP and GORP. Priority of Stock_Register is 14 in UCP, and 15 in GORP. Stock_Retrieve is 7 in UCP and GORP. Priority of Stock_Delete is 19 in UCP and GORP. Also priority of Sale_Register is calculated as 17 in UCP, 16 in GORP. Priority of Sale_Retrieve is 6 in UCP and GORP. Priority of Sale_Update is measured as 3 in UCP, 4 in GORP. Sale_Delete is calculated as 5 in UCP and GORP. Priority of Product_Register is 21 in UCP and GORP. Priority of Product_Retrieve is 12 in UCP and GORP. Priority of Product_Delete is calculated as 16 in UCP, 17 in GORP, and priority of Income_Retrieve is measured as 1 in UCP and GORP. Priority of Inventory_Retrieve is 10 in UCP and GORP. Priority of Expense_Register is 11 in UCP and GORP, and priority of Expense_Update is measured as 4 in UCP, 3 in GORP. Expense_Retrieve is 2 in UCP and GORP, and priority of Expense_Delete is 9 in UCP and GORP. Print is calculated 22 in UCP and GORP.As shown in a chart of Figure 3, the verification of Goal-oriented Use Case Requirement Priority has been done by using Use Case Points. The results are shown as almost in consistent although there is a little difference in ranking caused by the subjective difference of two methods.

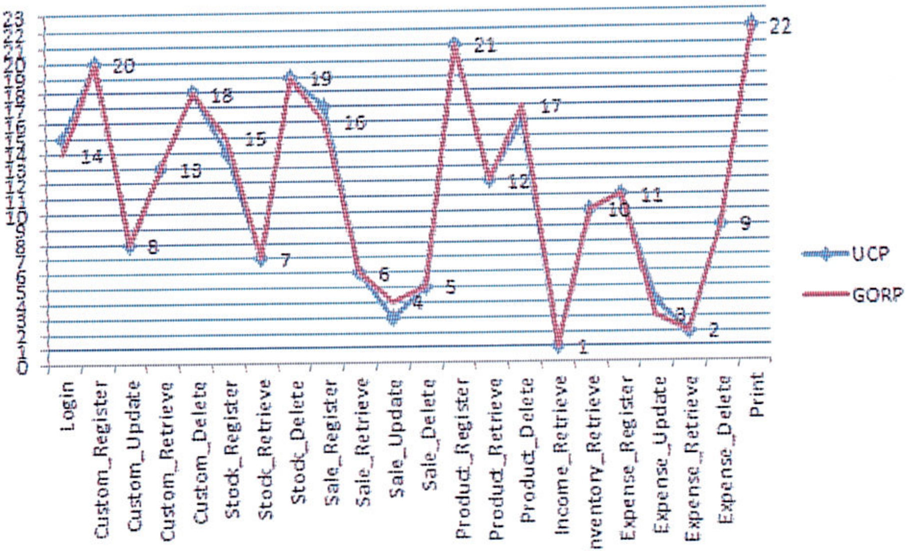


Figure 3. Chart for UCP and GORP [7]

6. Conclusion and Feature Work

In this paper, to extract the customer's requirements is using goal-oriented requirements process [7], and to calculate use case priority. Also priority is calculated by applying transactions and technical complexity factor using the Use Case Points [5].

We verify the priority of requirements based on the goal-based Use Case by comparing the Use Case priority calculated by the goal-oriented requirements process method and the Use Case priority using the Use Case points. As a result, though there is a slight difference between two priorities due to the subjective judgment of the evaluator, we conclude that the results are consistent with each other.

However, when we see the result of the Use Case priority using the Use Case points, it has a difficulty to calculate the Use Case priority due to the wide range of the technical complexity factors and environmental factors. Therefore, as our future work, it will be studied that research on finding the new technical factors and environmental factors which will be applied to the Use Case priority. Also we try to apply Use Case Points and Function Points to goal-oriented requirements process for the duration of project and the cost of project. Furthermore we compare and verify two results of methods.

Acknowledgements

This work was supported by the IT R&D Program of MKE/KEIT [10035708, "The Development of CPS (Cyber-Physical Systems) Core Technologies for High Confidential Autonomic Control Software"] and the Ministry of Education, Science Technology (MEST) and National Research Foundation of Korea(NRF) through the Human Resource Training Project for Regional Innovation.

References

[1] B. R. Grady, "An Economic Release Decision Model: Insights into Software Project Management", Proceedings of the Applications of Software Measurement Conference, Orange Park, FL, Software Quality Engineering, **(1999)**, pp. 227-239.

[2] J. Vijayan and G. Raju, "A New approach to Requirements Elicitation Using Paper Prototype", International Journal of Advanced Science and Technology, vol. 28, **(2011)**, pp. 9-16.

[3] K. Rhee, W. Jeon and D. Won, "Security Requirements of a Mobile Device Management System", International Journal of Security and its Applications, vol. 6, no. 2, **(2012)**, pp. 353-358.

[4] K. E. Wiegers, Software Requirements. MicrosoftPress, **(2003)**.

[5] G. Karner, "Resource Estimation for Objectory Projects", Objective System SF AB(copyright owned by Rational Software), **(1993)**.

[6] B. Park, S. Moon, D. Kim, C. Seo and R. Youngchul Kim, "A Study on Extraction of Goal Oriented Use Case Based Requirements", Proceedings of 2012 Korea Conference on Software Engineering, **(2012)**.

[7] B. Park, S. Moon, C. Seo and R. Youngchul Kim, "Extraction & Prioritization of Goal Oriented Use Case Based Requirements", Proceedings of KISM Spring Conference 2012, vol. 1, no. 1, **(2012)**, pp. 62-65.

[8] R. Bokyung Park and K. Youngchul, "Extraction & Prioritization of User Preference Requirements through User Needs", The 37th KIPS Spring Conference, **(2012)**.

[9] S. Y. Moon, B. Park and K. R. YoungChul, "Verification of Requirements Extraction and Prioritization using Use Case Points", Information Technology and Computer Science, ASTL13, **(2012)**, pp. 100-104.

[10] I. V. Koskosas and N. Asimopoulos, "Information System Security Goals", International Journal of Advanced Science and Technology, vol. 27, **(2011)**, pp. 15-26.

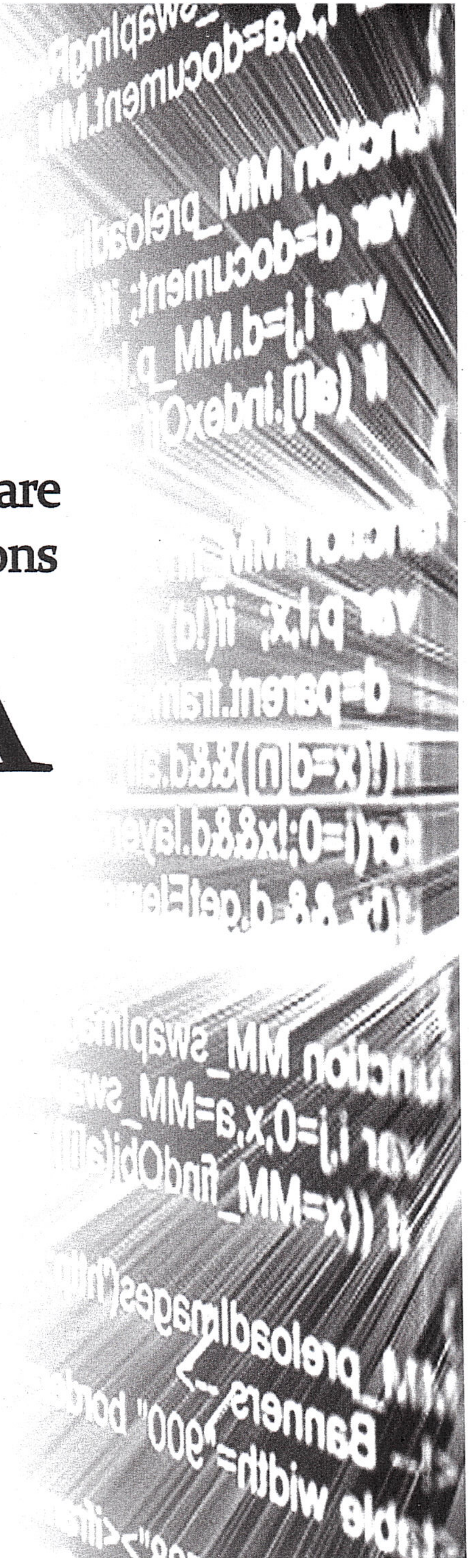
Authors

So Young Moon received her B.S. and M.S. degree in Software Engineering from Hongik University, Korea in 2007. He is currently a Ph.D. candidate in Hongik University. His research interests are in the areas of Use Case Point, Function Point, Goal-Oriented Requirements Process and Requirements Prioritization.

Robert Young Chul Kim received the B.S. degree in Computer Science from Hongik University, Korea in 1985, and the Ph.D. degree in Software Engineering from the department of Computer Science, Illinois Institute of Technology (IIT), USA in 2000. He is currently a professor in Hongik University. His research interests are in the areas of Test Maturity Model, Embedded Software Development Methodology, Model Based Testing, Metamodel, Business Process Model and User Behavior Analysis Methodology.

International Journal of Software Engineering and Its Applications

IJSEIA



SCIENCE & ENGINEERING
RESEARCH SUPPORT SOCIETY