Business System Analysis Model with Extended Entity Concept

AUTHORS :

Seojeong Lee

Dongduk women's University,

full time Instructor, Division of computer science,

sjlee@dongduk.ac.kr

Byungsun Ko

Sookmyung women's University,

Ph.D. student, Division of information science, Major in computer science

kobs@cs.sookmyung.ac.kr, kobs@orgio.net

Jainyun Park

Sookmyung women's University,

Professor, Division of information science, Major in computer science

TEL. 822-940-4599 / 822-710-9153

FAX. NO. 822-710-9296

Abstract

Existing system analysis models suggest various ideas to present real systems. However, there are no ideas to identify, arrange or drive definitely. To define and construct entities is a basic and important work of software development. And constructed entities can be major assets of business system. Existing analysis models implement rules or policies in business systems as program modules. In this case, whenever rule or policy changes, all relative designed or implemented outputs have to be changed.

In this paper, we suggest a business system analysis model to define, drive and present entities from real system to diminish problems above. This method has the properties to present rules as entities, to introduce management diagrams so that specify physical or administrative management items of entity and event, and to describe state transition diagram based on user interface. It is different from other models. The documents derived from this model are a context diagram, entity diagrams, event diagrams, information structure diagrams, detail event specifications, management diagrams and user interface state transition diagram. Through entity diagram and information structure diagram, we can guess real database structure.

1. Introduction

The analysis of system development process is important part to support design, implementation and management, and it can be the level to lead the most users' participation. The major works of this level are the analysis of information on system, user requirement, processes, legacy system status and management status. Among them, the basic information inside system is identified to change not often, so called static. The rests of them may be called dynamic, relatively. To manage system stable, static information must be constructed systematically. This implies that strong repository must be constructed, then the system can take out reliable response to any request. As the target system is bigger, the more systematic information must be constructed.

Several researchers introduce the guidelines to derive entities in system[1,2,3,4], but it is not specified how put information in order. As a result, after review of system analysis model presented complex, user cannot check whether his requests are normally applied or not. So the analysis model might be made excluded important information. This is the common drawback of ER analysis and OO analysis [5]. In some cases of OO analysis models, it is difficult to represent the whole system because so many entities are derived. To trim these, Yourdan suggested a method to identify candidate object. But, this method is not definite so each analyst can make each output [5].

This paper suggests a business system analysis model to define entities of system, derive systematic and represent them. Business system has more stable data, and more complex structure than the other domains. And, users of business system cannot know well anything out of their treat. But, existing analysis models show entire structure of system or subsystem so it is hard that users or job staffs can check whether the system is analyzed well or not, whether any other information have to be requested is applied well or not. Indeed, the result is an abundance of painstakingly crafted, highly detailed models that very few people can truly understand [6].

Today, because of progress of development environment as IT, time to build system has been increased. But because of changes of business circumstances - for example, EC, increased user requirements and system management manners, acceptable deployment time has been decreased. Figure1 shows it.

To overcome this unbalance, you can think to reduce system development time. The development time can

be reduced to change the hardware and software with preserving legacy system information. It implies that the application is just changed while information in system has been preserving. If basic information of system is constructed robust, changing system to new circumstances may be smoother. So this paper suggests a business system analysis model to identify basic information and derive entities.

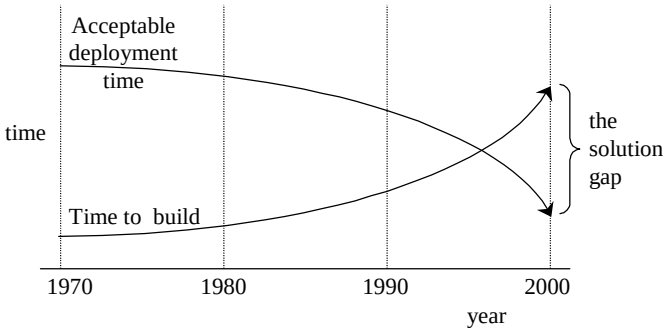

Figure 1 The expanding solution gap

2. Related works

Jacobson's interface object didn't suggest the method to view inside of object[7, 8], but this paper shows inside of interface as information diagram[9]. Jacobson's scenarios to specify event flows are similar to event diagram of this paper and control objects derived from scenario are similar to actions of action diagram.

The view of OO(object-oriented) model, for example Rumbaugh[10], or Booch[11,12], has similar view to the model suggested in this paper[13]. And, the method to derive entities is similar to ER model and Jacobson. OO model is different from ER model how to represent real world, but similar how to choose entities. However, the model we suggest is similar to OO model or ER model as the manner to derive entities, and different as the manner to identify, put in order and manage them.

Taylor[6] specifies that existed application-based business system has to change to model based. To preserve legacy system stable, three-layered structure must be constructed as in Figure 2. He partitioned the whole system to fractions and then gathered related fractions, called chunk. Figure 3 shows chunks that can be implemented to program module and encapsulated inside.

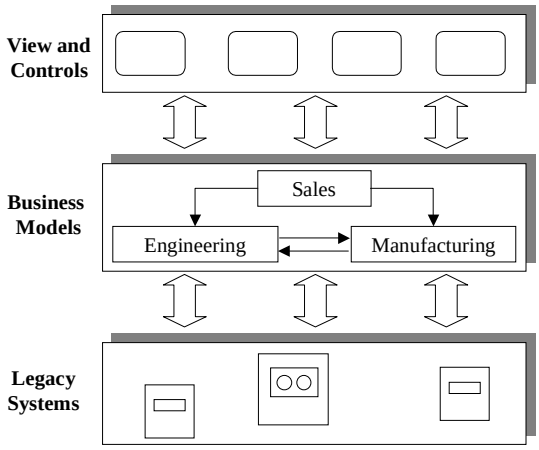

Figure 2 A layered architecture

UML suggests the definition and categories of object based on Jacobson's. However, it has some gap between analysis model and design model. If they cannot make narrow this gap, as the time goes, implemented system might be more different from desired system. So, deriving and constructing entities of the whole domain are the most important jobs of analysis level [3,14,15].

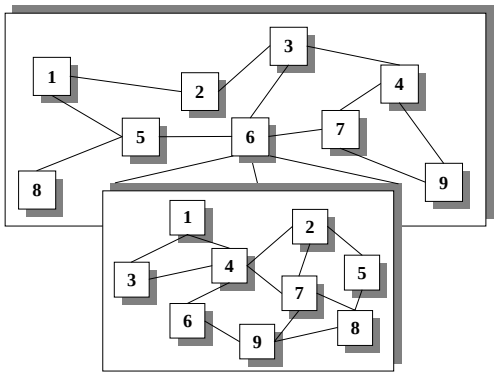

Figure 3 Designing with "chunks"

Catalysis approach for UML introduces “invariant” as the way to represent the rule of business model[20]. This idea is similar to our model, but we construct as entity and manage the history. For example, when it is

happened to be changed the rule, we can make its history safe as an entity, but Catalysis doesn’t suggest how to.

3. Information Structure Modeling

3.1 Basic concepts

This paper suggests "Information Structure Modeling" as an analysis model that preserves the legacy system supporting new environments and users' various requirements. The system makes responses to the users' requirements by using the information in itself. In this paper, "event" means a requirement and "entity" means the information of users, products, equipment and policies[16]. The "event" of the system can be solved by the cooperation of these entities.

First, the user information

This means the information of users who are using the system and purchasing the products managed in the system. For example, this can be a student in the course application system, a staff in the personnel management system, a resident in the administration system and a customer in the bank system. And it can also be the vendor as a product supplier, the business administrator, the information manager, and so on.

Second, the product information

This means the information of products to be managed in the system. For example, this can be a course of the course application system, a good of the department or a finance trade of bank.

Third, the equipment information

This means the information of making products or services. For example, classrooms can belong to it.

Forth, the policy information

Polices are so important information that they have more weight in the system when the systems are

managed. In the course application system, this can be the standard of compulsory subject, the registration condition of minor study, and so on.

The common feature of the above four is static, which is not less influenced by the change of environments of the system.

The "entity" of this paper is similar to that of the existing analysis models. But the viewpoint including information of policies is distinct from that. According to the concept of this paper, when the system deals with the "entity" to resolve the "event", it is also distinctive that the system deals with not the "entity" itself but "the view of entity". Existing models also deal with the "view" in the development level, but don't use the "the view of entity" in the analysis level.

The "event" is users' request to the system. The system executes the one or more "actions" to resolve the "event". Figure 4 shows the relationship between the "entity" and the "event" of the system. The user sends the "event" to request to the system[19]. Then, the system should execute the suitable "action" to solve the "event" given through the user interface. In the process of solving the "event", the "entity" constructed within the system can be used. And finally the system answers the user with the result of actions executed and coordinated.

Figure 4 represents the user, the entity (information of products, equipment, policy), and the management information by means of the entity diagram. In this case, the user is that of the context diagram. The events created in the system are represented on the event diagram. The entity and the view of the entity to solve the event on the event diagram are represented on the information structure diagram. The actions to resolve the events are represented on the action diagram. All the events of the system and the management information of the entities are showed on the management diagram. If the policy is contained in the program module, whenever the policy is changed or added, all the related program modules should be changed. To solve this problem, deriving the entity as the static information from the policy is suggested. Because the content of the

entity can be changed as time passes, time concept is introduced to the entity diagram in this paper.

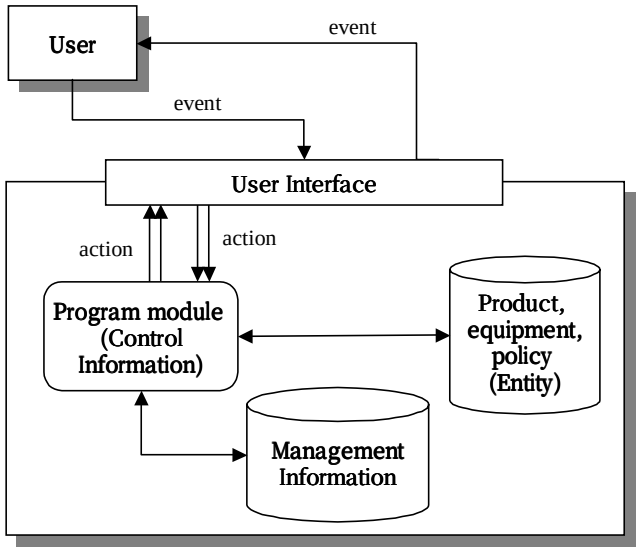

Figure 4 The relation of information

It is obscure to define the state from the state transition diagrams of the existing analysis model. So, even though the analysts use the same method, the result may be often quite different. For example, some are made in the view of the user, others are made in the view of the process or the data. Therefore, many the state transition diagrams with different viewpoints can be created in the same system. In this paper, we suggest the user interface state transition diagram that represents the user interface as one state.

3.2 systematic derivation of the entity

In the system, there are information of users, equipment, products, policies and management managing the information. All this information not only takes important role in operating the system but also become property of the business system. So, the work of constructing the information systematically must be very important thing to consider in the analysis level.

The entity is extracted from the view of the entity in the information structure diagram, the management information in the management diagram and the information that the system has basically about the user, the

equipment, the policy and so on. Figure 5 shows the relation between the information of the analysis model and the structure of the entity. The three dimensional entity means that the change of the entity as the time passes would be logged. For example, the entities from the management diagram and policies should be presented by the three dimensional block. This means that the whole management information of an entity of the management diagram is contained in one block. In addition to, though the information of equipment, products and policies are not presented as the diagram through the analysis level, they can be created as the entity. To manage these entities, it is necessary to manage the configuration management of the entity, the effect-scope management, and write-protection of the information of the entity

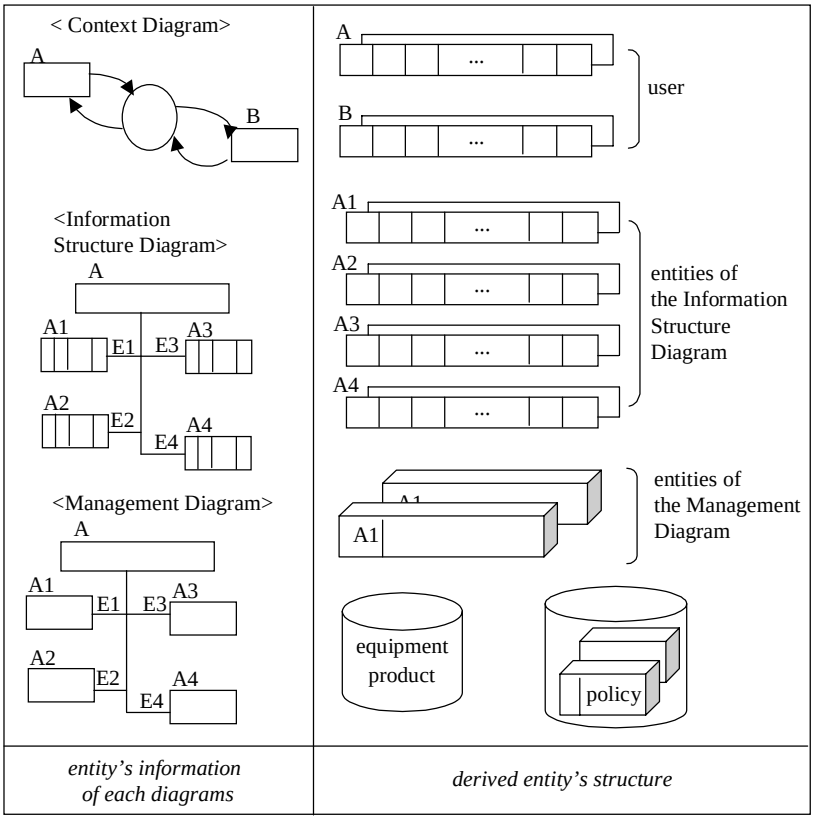

Figure 5 The example of entity structure derivation

3.3 The analysis model

(step 1) Draw the context diagram.

The context diagram represents the scope of the system. From this, what process will happen in the system can be recognized[8]. In this diagram, the ellipse means the system, the rectangle means a relator. By means of the arrows between the system and the relater, "event" and "response" of the system are represented. A Relater is a user who can give an access or stimulus to the system directly. The user is identified to the client and the job staff. Figure 6 shows these concepts of the context diagram.

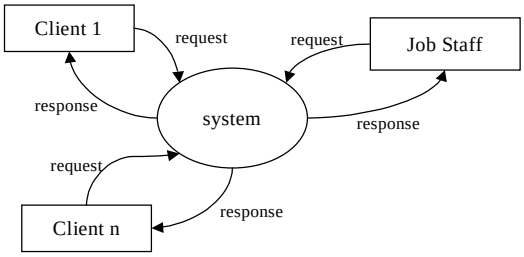

Figure 6 The context diagram

(step 2) Specify events of users in event diagram.

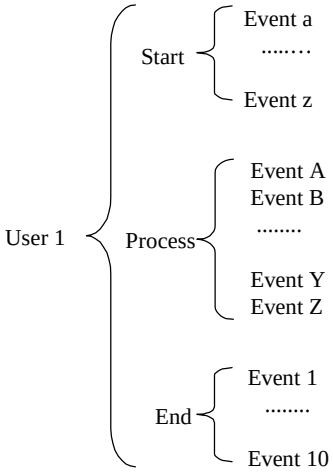

Figure 7 The event diagram

The event diagram shows events that a user request to the system in the form of WOD(Wanierr Orr

Diagram)[17]. WOD is descriptive expression of event processing. It is made up of three parts, "start part", "process part", "end part", such as Figure 7. And it describes in detail relation between the user of the context diagram and the system. The context diagram establishes the scope of the system. The event diagram describes activities of the user which is made as many as the users.

In WOD, the brace({ } mark is identical with the left mark of the set. Put down the user name at the left part of the brace. The events happening when the user makes an approach to the system are written in the start part of WOD. In the end part of WOD, the events happening when the user finishes the use of the system are written.

(step 3) Specify the detail event specification

The detail event specification is used to specify actions by which an event is processed. Actions are to be some precondition before process an event, some postcondition after process an event or simply a part of an event. As event diagram, single brace({} is used. precondition is specified as “start” part, and postcondition as “end” part. The process of actions is sequential basically, and in the case of one of many, you can use ⊕ , and logical AND or logical OR is represented “AND” or “OR”, respectively.

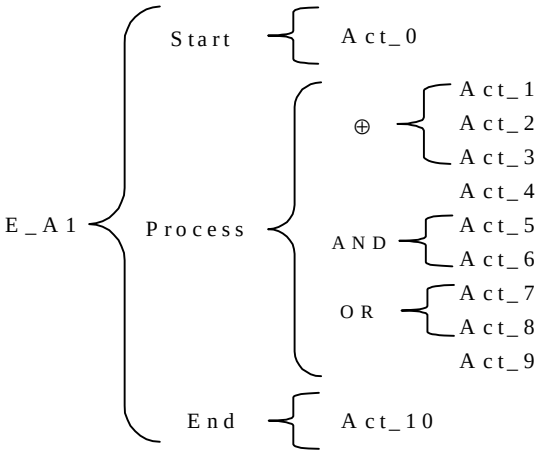

Figure 8 The detail event specification

Figure 8 shows an example of the detail event specification to resolve event E_A1. The sequences of

actions are several ways by cases. Act_0 is precondition , Act_10 is postcondition of E_A1. So we can think sequences as follows:

1) If Act_0=OK, then Act_1 -> Act_4 -> (Act_5 AND Act_6) -> (Act_7 OR Act_8) -> Act_9 -> Act_10

2) If Act_0=OK, then Act_2 -> Act_4 -> (Act_5 AND Act_6) -> (Act_7 OR Act_8) -> Act_9 -> Act_10If Act_0=OK, then Act_3 -> Act_4 -> (Act_5 AND Act_6) -> (Act_7 OR Act_8) -> Act_9 -> Act_10

3) If Act_0<>OK, then EXIT

(step 4) Specify the information structure diagram

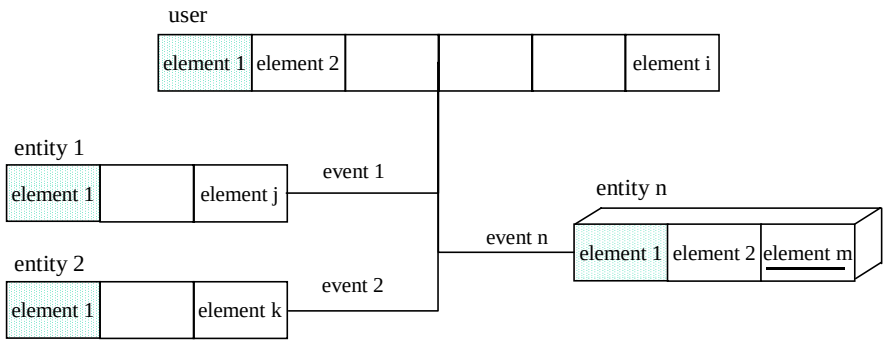

Figure 9 The information structure diagram

The ISD(information structure diagram) represents the information to resolve the events, such as Figure 9. The analyst can find out that we can make any request to the system and any information should be prepared to resolve the events by means of the ISD. Each part of the ISD is filled up the attribute name. And then, in design or development level, these are to be conceptual database schema. Of course in real database, it should be necessary to redundancy check, normalization, database organization strategy, and so on. The user can recognize information that is requested to events, and the developer can recognize conceptual database schema by the ISD[18]. Each part with shadow of the ISD is primary key of the entity. The attribute name with a underscore(_) represents “linking entity” which is referencing other entities. The entity referencing other entities is located in either the same ISD or the different ISD. And, the descriptions of them are

represented in the management diagram.

(step 5) Specify entity diagram

Information structure diagram shows some entity information, which is a view of an entity used for event processing. Even if a user refers same entity name for two events, each view for each event can be different, so we introduce entity diagram to show inside of real DB. Some entities of information structure diagram can be made up as one entity. And, each user of information structure diagram must be a entity of entity diagram. In Figure 10 and Figure 11 show this situation.

In Figure 11, entity A and B represent users of Figure 10, and entity A1 is merged from entity for event E_A1 and event E_B1 of Figure 10, so that has 8 attributes. In this manner, entity diagram is going to be increased to completion of analysis model.

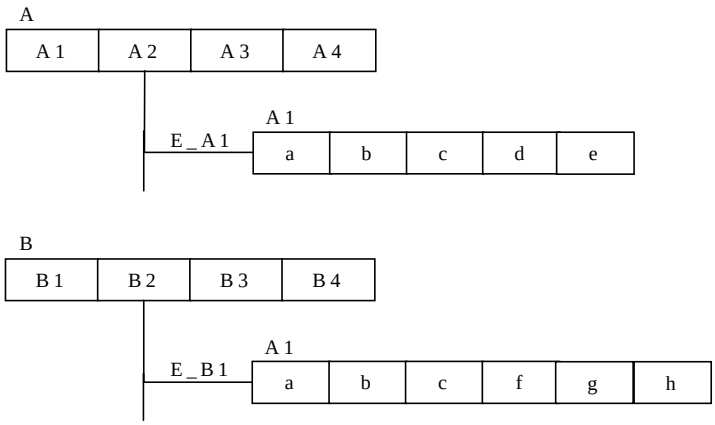

Figure 10 The example of information diagram

(step 6) Specify management diagram

Management diagram shows physical information about conceptual information of formal diagrams, for example, authority, physical DB location, access level, DB name or table name, etc. Figure 12 shows it. Each rectangle corresponds to each entity of information diagram. We think definite items as follows and because it will be referred implementation level.

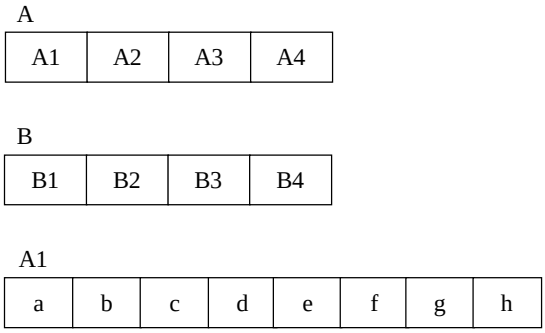

Figure 11 The example of entity diagram

- manger name

Specify who is the manager of DB or table. DB manager or job staff might be.

- authority level

It implies level defined in system as status or job. When any user tries to log-in, system refers this level so that can restrict to available task

- DB name and table name

Real DB and table names are written.

- linked table

If a attribute has to be referred any other table, we call it “linking entity” and represent with underscore(_) at information diagram.

- program module name

A event is implemented as one or more program module or a part of user interface. So we can specify these program module(s) name or user interface name.

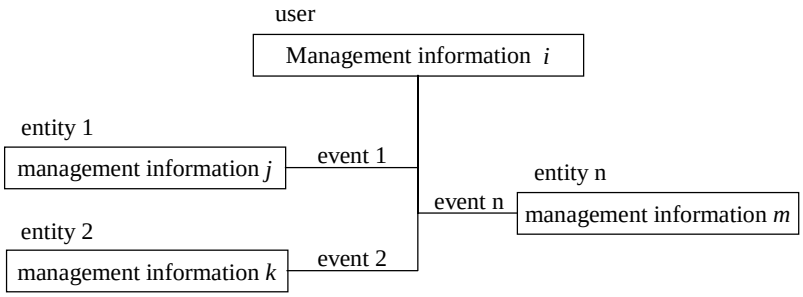

Figure 12 The management diagram

(step 7) Draw interface state transition diagram

Interface STD can help to understand system flow. Figure 13 is a interface STD of ATM. It is similar to other models' STD, but it is different clearly to take a state as a user interface. In addition, synonym table, attribute default value table can be made out.

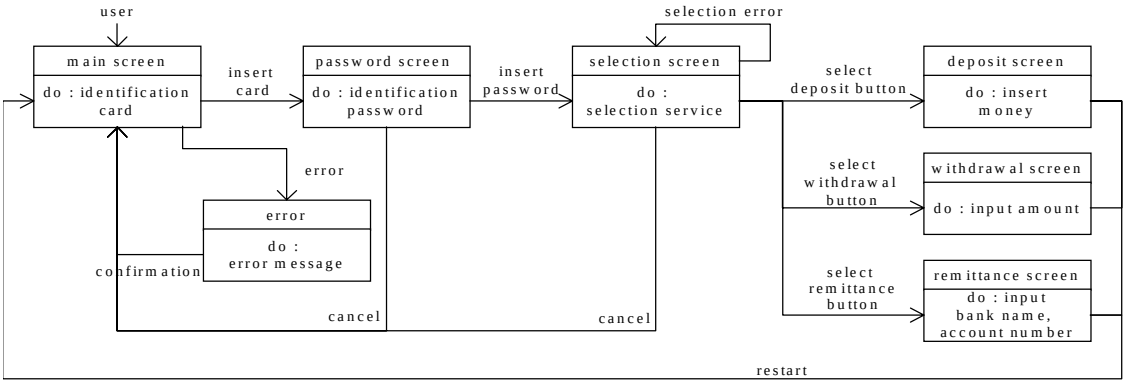

Figure 13 The information transmission structure of ATM

4. Properties

4.1 Policy entity

Though existing model can derive entities and events from the policy, but it cannot be treated as entity and just implemented as program module at implement level. In this case, to involve the changes of policy, every program module contained related policies must be updated. This makes development time longer and the

cost higher. To reduce these drawbacks, this paper suggests that the rule is derived as entity that the changes of rule can be accomplished as just DB update. This can lead to diminish the overloads of program implementation and management cost.

4.2 The Management diagram

This makes help users or designers to understand the whole system management. Detailed items are administrative information, security and physical location information, networking site information, etc. It is implemented as databases or tables in repository. Even though any other analysis models don't take, it might be an important role to manage entities systematic.

4.3 User interfaces STD

STD is a tool to represent system flow dynamically so almost analysis model take it. Existing STD presents state transition between processes or events [3,10]. But, there is no define about which is a state, then analysts have some chaos to make STD. We suggested User Interface STD to define a state as a user interface. With this STD, user can understand visible system flows and analysts can take it a tool to communicate with users.

5. Conclusion

As an effort for stable business system analysis, we suggested a business system analysis model to define entities four types and construct them. Stable construction of entities gives the system to adapt easily to the change of internal or external environment. And, to derive policies as entity has an effect to reduce system management cost.

According to our concept and notation, we can start partitioning analysis so that can increase more and more. This takes us higher system reusability and it can be adapted to e-biz, distributed system, or EC system.

The simple notation can take a chance to participate aggressive for users who are not close to computer system. So users - job staffs and clients – might have new ideas to request or response more efficiently.

References

[1] S. Shlaer and S. J. Mellor, The Shlaer-Mellor Method, Project Technology Inc., 1996.

[2] Donald G. Firesmith, Object-Oriented Requirements Analysis and Design, John Wiley & Sons, 1993.

[3] Martin Fowler, UML Distilled Applying the Standard Object Modeling Language, Addison-Wesley Longman Inc., 1997.

[4] Rational Software Co., UML ver. 1.0 Notation Guide, Rational Software Co., 1997.

[5] E. Yourdon, Object-Oriented Design, Prentice-Hall, 1994.

[6] David A.Taylor, Business Engineering with Object Technology, pp. 13-28, John Wiley & Sons, 1995.

[7] Ivar Jacobson, The Object Advantage : Business Process Reengineering with Object Technology, Addison-Wesley, 1994.

[8] Ivar Jacobson, Grady Booch, James Rumbaugh, The Unified Software Development Process, Addison-Wesley, 1999.

[9] SeoJeong Lee, A object-oriented system analysis and design supporting for user-based consistent view, Ph.D. Thesis from SookMyung Women's university., 1998.

[10] James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, William Lorensen, Object-Oriented Modeling and Design, Prentice-Hall, 1991.

[11] Grady Booch, Object-Oriented Design, 2nd Ed., Benjamin Cummungs, 1991.

[12] Grady Booch, Object Solution: Managing the Object-Oriented Project, Addison-Wesley, 1996.

[13] The Object Agency, A Comparison of Object-oriented Development Methodologies, The Object Agency Report, 1995.

[14] Doug Rosenberg, Kendall Scott. UML Object Modeling, Addison-Wesley, 2000.

[15] JeongIn Jeong, A study on the object extraction methodology using the information structure modeling in the business work, M.A. Thesis from SookMyung Women's university., 2000.

[16] JaeNyon Park, "System Analysis by Information Structure Diagram", a collection of thesis in

SookMyung women's university, Vol. 33, 1992.

[17] JaeNyon Park, Fundamentals of Computer Science, pp. 337-366, KyoHakSa, 1997.

[18] HyunHee Kim, Object extraction and analysis of the scope of effects on modification in information structure model, M.A. Thesis from SookMyung Women’s university., 1999.

[19] Grady Booch, James Rumbaugh, Ivar Jacobson, The Unified Modeling Language User Guide, Addision-Wesley, 1999.

[20] D'Souza and Wills, Object, Components, and frameworks with UML: The Catalysis Approach, Addison-Wesley, 1999.