DIAGRAMMATIC EDUCATION FOR SOFTWARE ENGINEERING

Ryo Kawabata
Kiyoshi Itoh
Laboratory of Information and Systems Engineering, Sophia University, Tokyo, Japan

In the education of software engineering, it is important to educate learners about diagrammatic representations for analyzing configurations or functions of a target system. In systems analysis two or more diagrammatic representation are described in appropriate order, which experts know as the effective analysis process. Education of the analysis process using diagrammatic representations is more important and essential rather than education of notations of diagrammatic representations. This paper proposes an intelligible analysis process which is suitable for education. Learners of systems analysis can acquire the process model effectively by using the process model, which includes Petri Net, STD, DFD, and IDEF0, in this order. It is easy to reuse components between the diagrammatic representations according to the analysis process.

Keywords: *diagram, education, software engineering.*

1. Introduction

In the education of software engineering, it is important to educate learners about diagrammatic representations for analyzing configurations or functions of a target system. Education of the analysis process using diagrammatic representations is more important and essential rather than education of notations of diagrammatic representations. This paper proposes an analysis process which is suitable for education. By using the process model, learners of systems analysis can acquire the process model effectively.

In order to analyze a system sufficiently, the target system is analyzed in terms of two or more viewpoints. As the system analysis advances, the different kinds of diagrammatic representations are described. Each diagrammatic representation shows a different view point (Kumagai *et al.*, 2004). The diagrammatic representation is not described from scratch but described by using components in a diagrammatic representation which is described in the previous phase of the analysis. By using diagrammatic representations in appropriate order, the system analysis is easy to understand and performed effectively since the reusable components can be found easily (Kawabata *et al.*, 2005). Experts know the combination of diagrammatical representations and the appropriate order of using them as an effective analysis process.

Also in the education of software engineering with diagrammatical representation, there is an effective analysis process. By analyzing the target system according to the analysis process, essential components can be found easily. Although the viewpoint is different depending on a diagrammatic representation, some components of the target system are described across two or more diagrammatic representations. Finding these common components facilitates the analysis in describing the target system.

The authors propose an analysis process using four kinds of diagrammatical representation such as Petri Net, State Transition Diagram, Data Flow Diagram and IDEF0. The authors apply the analysis process for teaching a class of systems analysis. Students in the class analyze an example system and their own target system using the analysis process.

2. Related Works

The paper (Johansson *et al.*, 1995) proposes a practice driven education program with the aim that the students are trained to learn by independently. Their approach promotes the active learning process for students, as opposed to simply teaching theories. This approach is similar to our class. This approach is similar to our class. In our class, most of the time is allocated to exercises that teach how to develop diagrammatic representations. This is effective for learners to acquire the ability of system analysis.

In (Boehm *et al.*, 1998, Mayer, 2001, Bagert *et al.*, 1999) and (Bourque *et al.*, 1999), the guideline and curriculum for software engineering education are discussed. In (Boehm *et al.*, 1998), the author tried to teach students software engineering by using the WinWin spiral model, UML and other models. The research (Mayer, 2001) discusses the software curriculum and argues that the software curriculum should include principles, practices, applications, tools and mathematics involving documentation and user interaction. Describing diagrammatic representations is one of the methods of documentation. In the class, students describe diagrammatic representation with two roles such as users and analysts. In (Bagert *et al.*, 1999) and (Bourque *et al.*, 1999), the guide line for software engineering is proposed. These guidelines organize essential knowledge areas and curriculum contents for software engineering. These are important for educating specialists such as a particular type of system engineer. Not only system engineering specialists but also system users should know how to conduct system analysis, without necessarily knowing everything a system engineer should know. It is important for them to communicate with developers about the target system. Our analysis process is suitable for them and beginners of software engineering to learn software engineering.

3. The Contents of the Class

The class is held in the second semester. About 40 students take the class. Two teachers and four teaching assistants support them to practice system analysis. In order to provide explanations on diagrammatic representations, the authors use a library management system as an example system. The authors explain the notation of diagrammatic representations using the example system. It is effective to explain the notation of diagrammatic representations by familiar examples. The library management system is a familiar example for students because they can use the system on campus or in the neighborhood. The other salient feature is that the system is simple and easy to understand. It is effective for explaining diagrammatic representation to use the same example across all diagrammatic representations. Using the same example, it facilitates the location of essential components of the target system and understanding the difference of viewpoints that diagrammatic representations might have.

The library management system can be broadly divided into three functions such as lending of books, retrieval of books and return of books. The authors explain the diagrammatic representation using one of the functions. Students practice analysis of the system by describing diagrams for the other two functions. After the practice, the authors show the examples of the diagram for the other two functions. Students check their analysis result referring to the examples. In their practice, teachers and teaching assistants go around students, and give some advice and take questions.

The authors prepare the list of example systems for students. These are simple systems, but students can refine their ability to analyze systems. Students select one target system from the example systems. About these examples, the system name and brief requirements are shown for students. Students start

requirements analysis for the target system which students select. Since the details of requirements are defined by each student, they are different, depending on the student. The extension of the requirements can be done freely.

1. Video/CD rental system
2. Airplane ticket purchase and boarding management system
3. Ticket purchase and entrance management system for a movie theater
4. The network system by home electrical products: Remote control of home electrical products
5. Information system for real estate
6. Sales management and stock control system for convenience store
7. Management system: course registration and issuance of a list of records
8. Information system for hotel: room reservation, check-in and check-out.

In the class, four types of diagrammatic representations such as Petri Net, State Transition Diagram, Data Flow Diagram and IDEF0 are used. In the analysis with Petri Net, personnel who give services to a customer, the flow of the transaction (customer), the order of activity, the relationships between personnel and activity, the exchange of information or material between personnel are clarified. In the analysis by State Transition Diagram, the state transitions of each customer, personnel, information and material are clarified. In the analysis by Data Flow Diagram, data flow between personnel and customer is clarified. At the last, in the analysis by IDEF0, the work flow of the system with strict semantics is described.

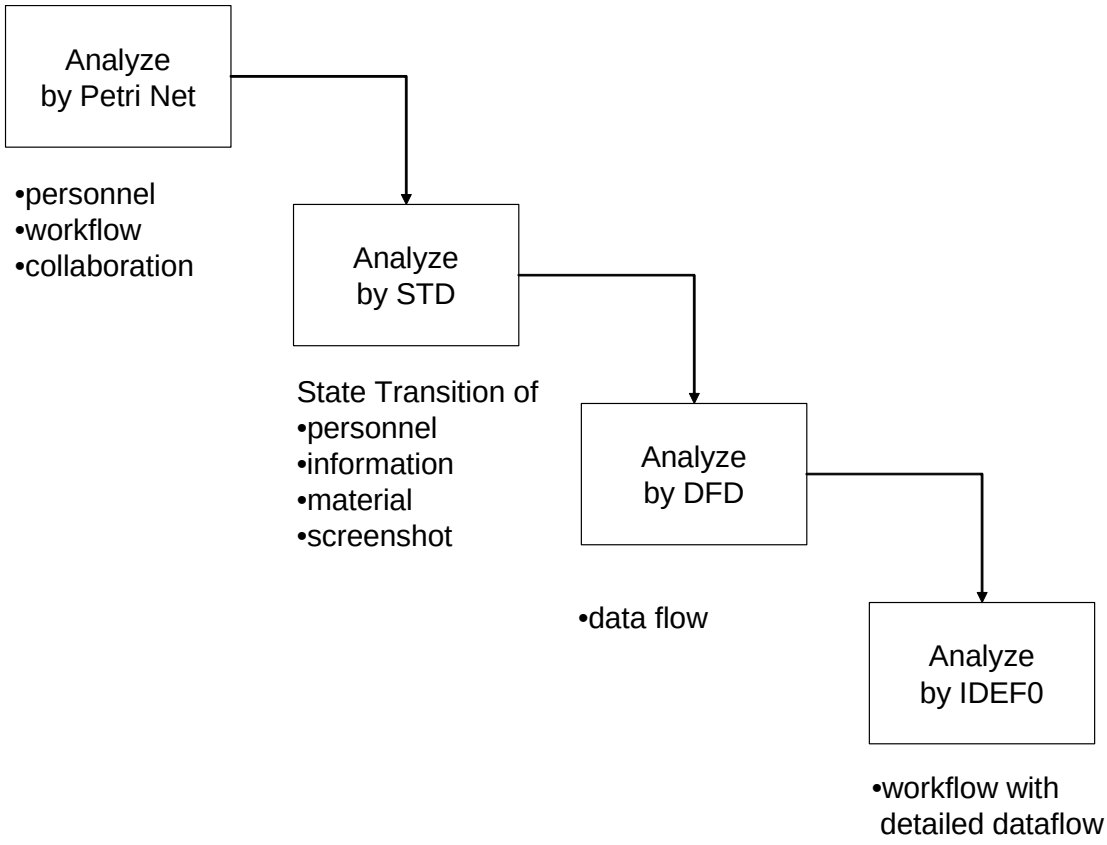

Fig. 1 Analysis process for education.

4. Petri Net

In an ordinary Petri Net (Peterson, 1981), the target system is not limited and the connecting rule is simple. By using the Petri Net, the system can be described freely, but the meaning of elements and relationships tends to be different depending on the analysts. The authors introduce discipline into Petri Net in order to apply Petri Net to analyze of the collaboration task (Senuma *et al.*, 2003). The target systems of the class are mainly information systems in which the collaboration task is observed. These systems are not special, but general. In the Petri net, personnel, workflow of each personnel and exchange between personnel are analyzed and described. At first, a customer who receives service form the target system is identified and described with places and tokens. Then the work flow of the customer is described with transitions which mean tasks, places which mean buffers of the tasks and arcs. Then the personnel who give services to the customer is identified and described with places and tokens. A task in the work flow is performed by the personnel, the transition which means the task and the place which means the personnel is connected by arc. If two or more personnel perform in order to give services, the work flows of each personnel are described. In this case, material or information is exchanged between these personnel, so the place which represents material or information is placed between the two workflows and connected to a task of each workflow.

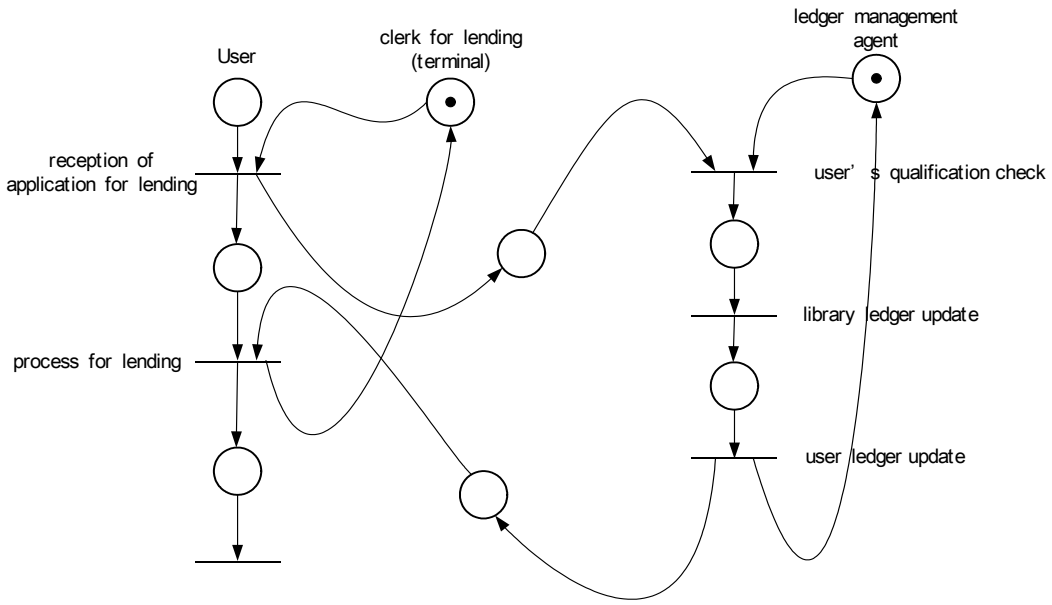

Fig. 2 Petri Net.

Figure 2 shows an example of a Petri Net. This Petri Net describes the lending process in a library. The flow on the far left represents the workflow of a customer.

In our class, a Petri Net is an important diagrammatic representation, since Describing diagrammatic representations in the following process starts by using components in a Petri Net. This method makes it is easy to describe and analyze the entire process of the system. Understanding of the behavior of a token such as waiting, competition and synchronization is a key point for describing a well-developed Petri Net. It is difficult for learners to understand the behavior of a token in a Petri Net described on a sheet. The authors use Petri Net Visualizer (Kawabata *et al.*, 2004) shown in figure 3 to show the behavior of a token.

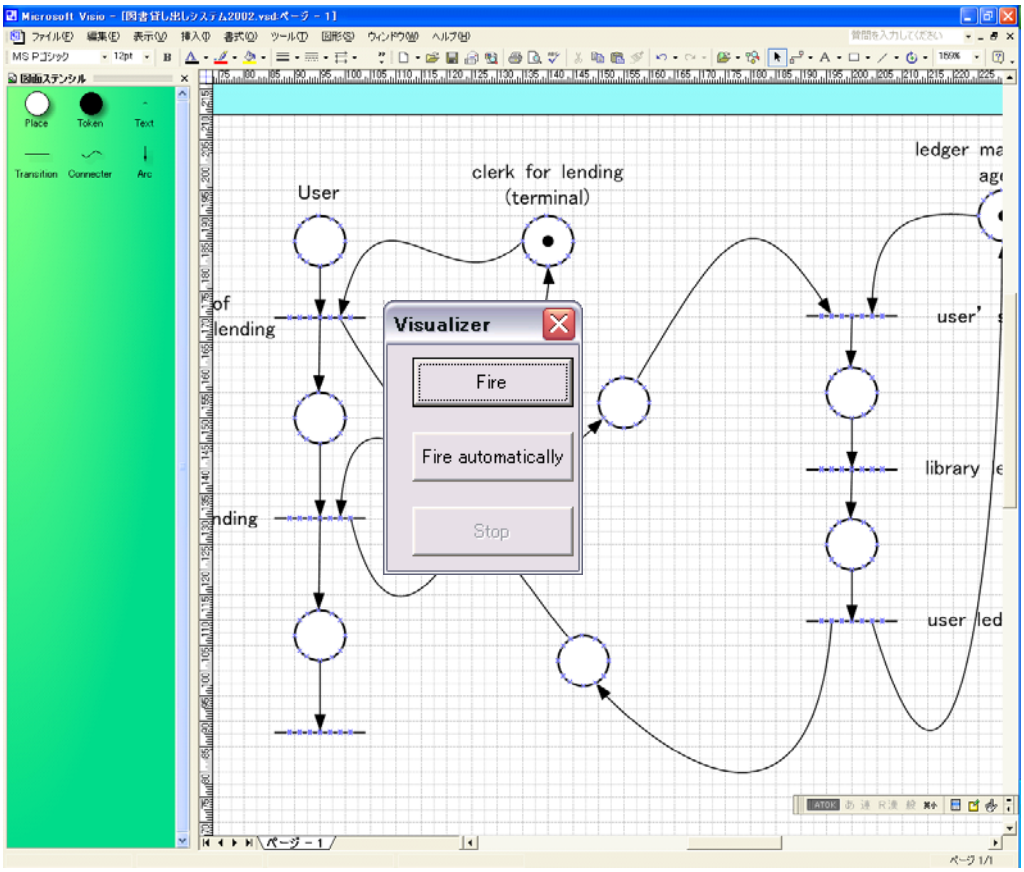

Fig. 3 Petri Net Visualizer.

5. State Transition Diagram

In the State Transition Diagram (Harrel *et al.*, 1990), two kinds of state transition diagrams are described. In the first diagram, possible states of personnel, information and material are identified and described. In the second diagram, state transitions for the screen user interface are described.

The first diagram is described for personnel/equipment and information/material. Personnel/equipment which gives service is described explicitly in the Petri Net as a place. The clerk for lending in Figure 2 is an example. Information/material is described implicitly in the Petri net. The term which represents or relates to information/material is included in a transition which is connected to a place representing an exchange of information/material. One STD is described for each personnel, equipment, material and information. The label of transition in a Petri Net can be used for describing state in a STD; for instance, the state “be receiving application for lending” is described from the label “reception of application for lending” of a transition in a Petri Net. Figure 4 shows an example of State Transition Diagram for a clerk in library management system. This diagram describes state transitions of the whole system. In order to describe the system in detail, a State Transition Diagram for each material, information or personnel is described.

After the first STD, the second STD for the screen, which is used for analyzing a user interface, is described. Figure 5 shows an example of a STD for the screen. After describing State Transition Diagram of screen, the graphical user interface is designed by hand. In this class, the implementation of the target system is not done, because time is lacking for teaching application programming.

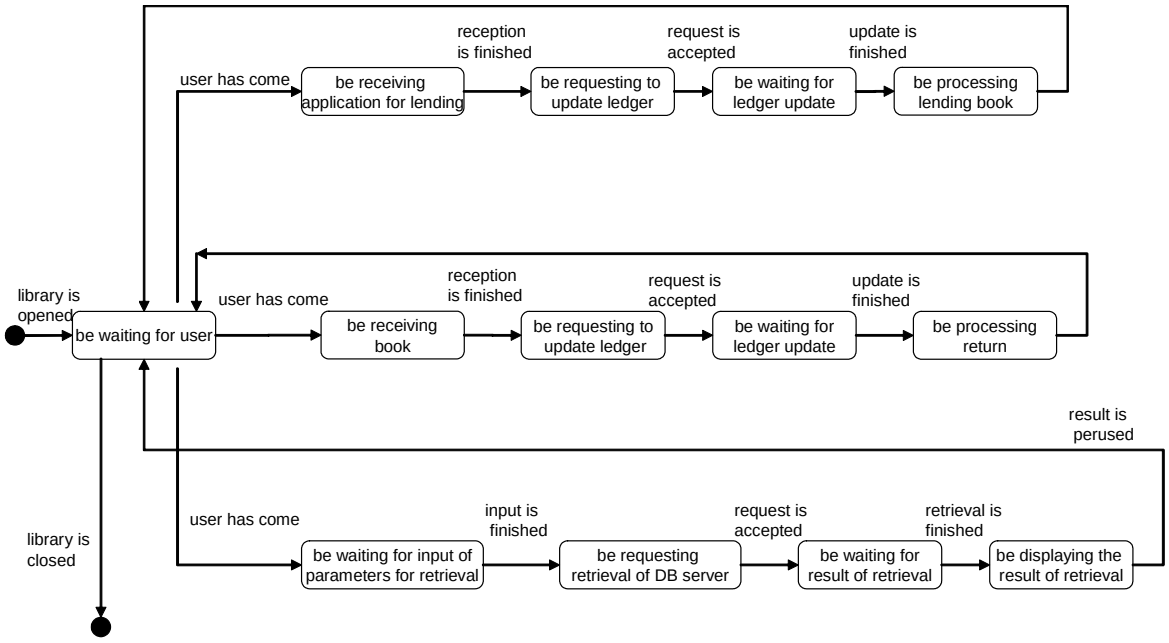

Fig. 4 State transition diagram.

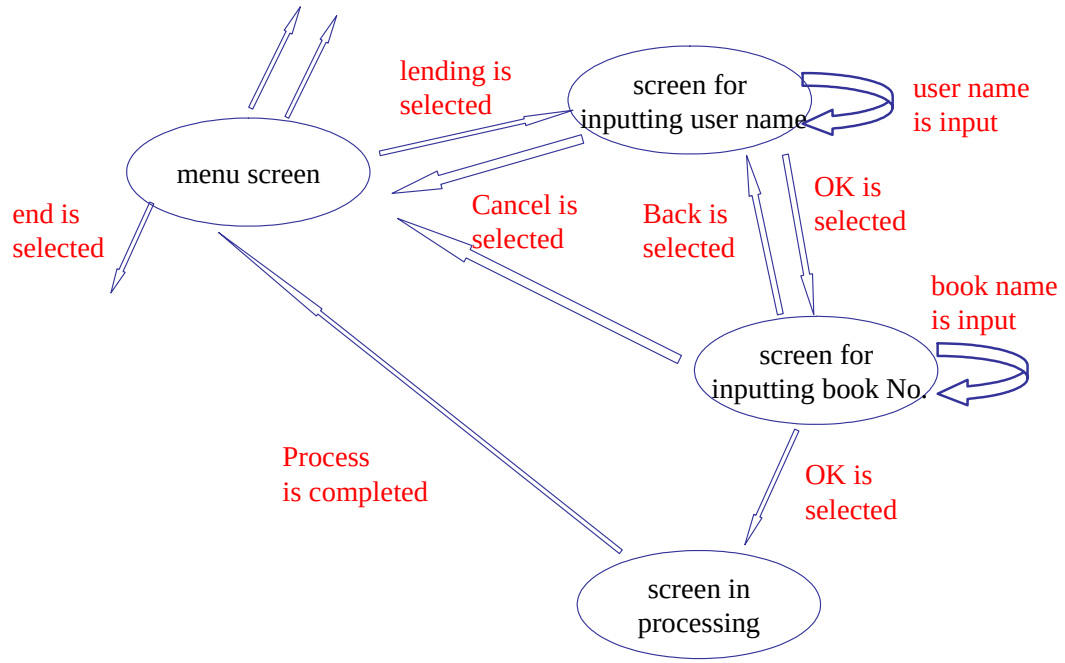

Fig. 5 State Transition diagram for the screen.

6. Data Flow Diagram

In a Data Flow Diagram (DeMarco, 1979), the flow of data in the target system is analyzed and described. The data flow diagram is described for each process such as lending, returning and retrieval. The components of DFD such as process, source/sink and arc are identified in a Petri net and a STD.

- The customer and personnel in a Petri Net are described as a source or a sink in DFD elements.
- The possible states in a STD are described as a process in a DFD.
- Arcs among processes, a source and a sink are described referring to arcs in a STD.
- The label on an arc is added manually by an analyst.

Figure 6 shows an example of a DFD. This shows relationships between processes and personnel in lending.

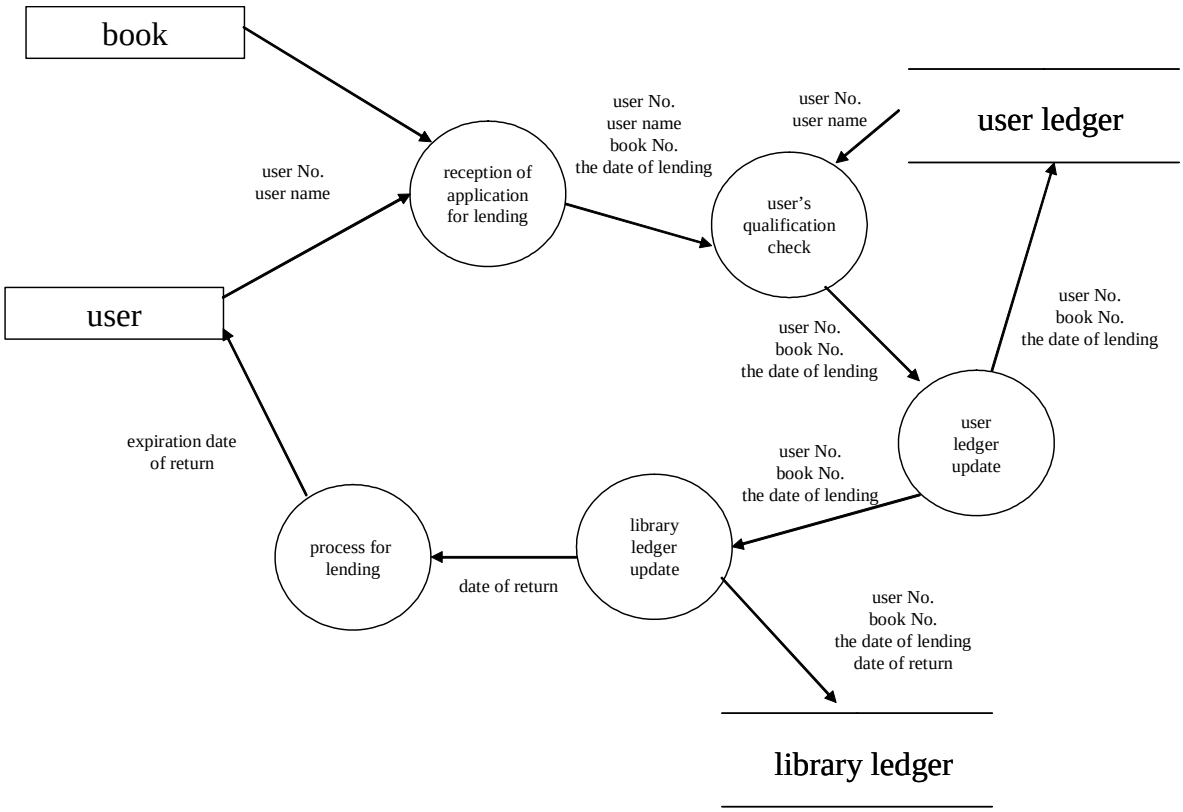

Fig. 6 Data flow diagram.

7. IDEF0

In IDEF0 (Marca *et al.*, 1988), workflow from an entire viewpoint is analyzed and described. An IDEF0 is described for each process such as lending, returning and retrieval. The components in an IDEF0 are identified in a Petri Net and a DFD.

- A process in a DFD is described as an activity in an IDEF0

- An order of activities is according to an order of processes in a DFD.
- Basically, an arc between processes in a DFD is described as an arc from output to input of an activity in an IDEF0
- A mechanism in an IDEF0 is described referring to a Petri Net. An activity in an IDEF0 is correspondent to a transition in a Petri Net. In a Petri Net, a transition is connected to a place which represents personnel who perform the transition. By this relationship, the mechanism in the IDEF0 can be identified.
- A start condition such as "task A is finished." etc. is described as a control. A control and a mechanism are added by an analyst.

Figure 7 shows an example of IDEF0. This shows a flow of lending book in a library.

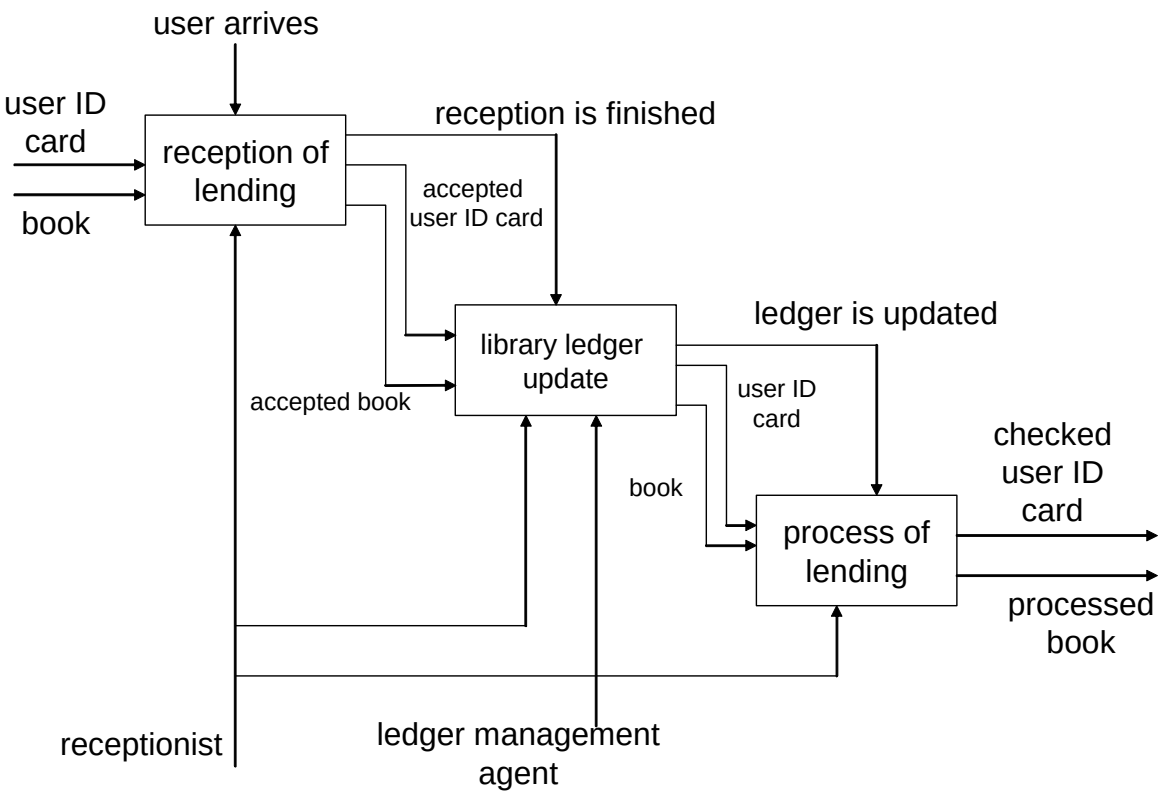

Fig. 7 IDEF0.

8. Database and E-R diagram

The authors employ Microsoft Access for learning a database, because the Access is installed on all PCs in our university. As an example of a database, a table of students, records of subjects and subjects are used. Figure 8 shows the relationships between the three tables defined with Microsoft Access. The example is easy to understand for students. In the class, a teacher explains the usage of Access, the defining method of tables and queries by using examples. After the explanation, students define queries

for practice problems. Teachers and teaching assistants guide students to define tables and queries. The queries prepared as practice problems are as follows.

- a list of students who takes score greater than 60
- maximum score for each subject
- average score for each subject
- a list of students who takes score greater than average score
- the number of students who takes score greater than average score
- a list of student's grade
- a list of GPA

After using Microsoft Access, students define database for their system. They define tables and queries based on their own idea. Then they describe E-R diagram. In the usual study of systems analysis, E-R diagram is taught before using database, but the authors teach in a retrograde order. In the case that the database is taught in ordinary order, students tend to describe database as an entity. By using a database, they understand it is wrong.

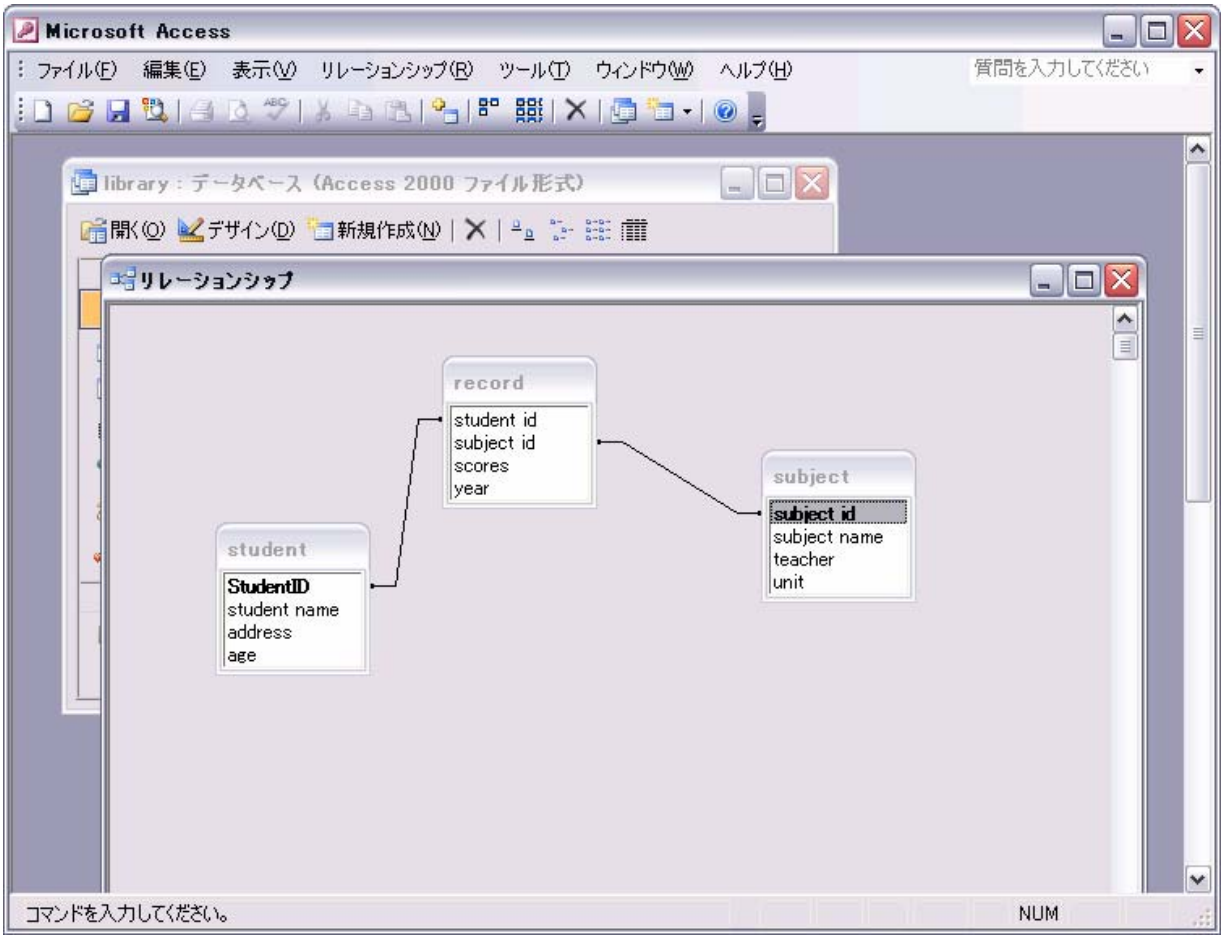

Fig. 8 ACCESS.

9. Examples

In our class, students describe several diagrammatic representations by using the proposed analysis process. One of examples is described in this section. Fig.9-14 shows the Ticket purchase and entrance management system for a movie theater. These diagrammatic representations are described by a member of the class.

Fig. 9 shows a part of a Petri Net which describes the flow of a customer and workflow of a terminal and a register. Fig. 10 and Fig. 11 show STDs for a customer, a register staff and a counter staff. These STDs are described based on a Petri Net. Fig. 12 shows a STD of screen which represents a purchasing process with a terminal. Fig. 13 shows a DFD which is described by using components in a Petri Net and STDs. Buffers such as seat information and cinema information, and data are added by analysts. Fig. 14 shows IDEF0 which are described by using components in diagrammatic representations described in previous phases.

Although some mistakes exist in these diagrammatic representations, the student who is a beginner of software engineering can describe the target system by using the analysis process.

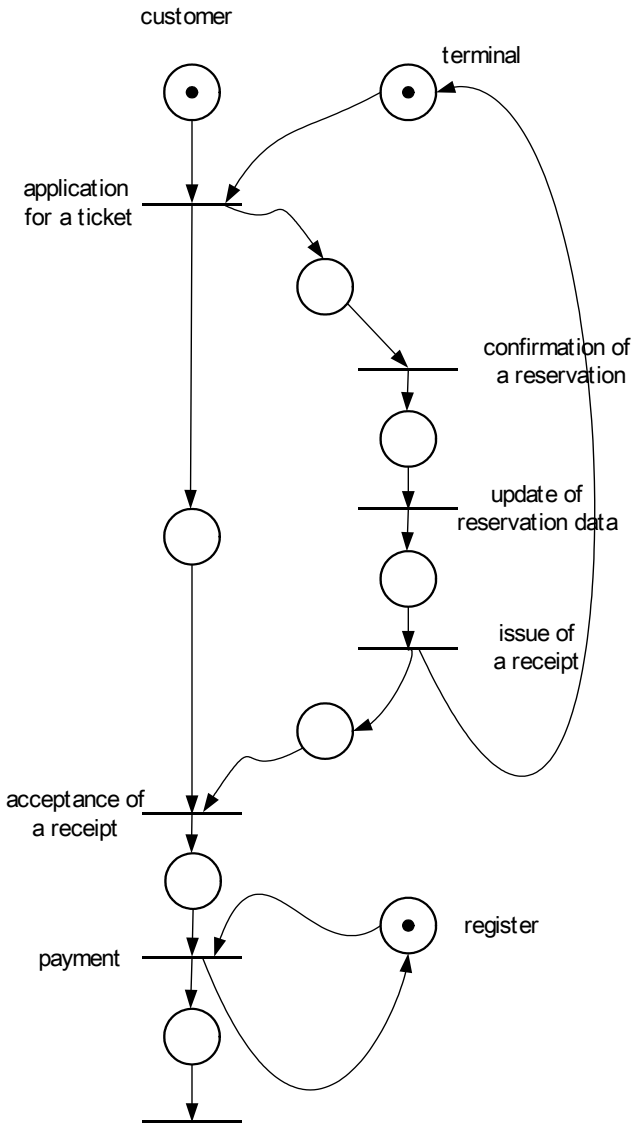

Fig. 9 Petri Net.

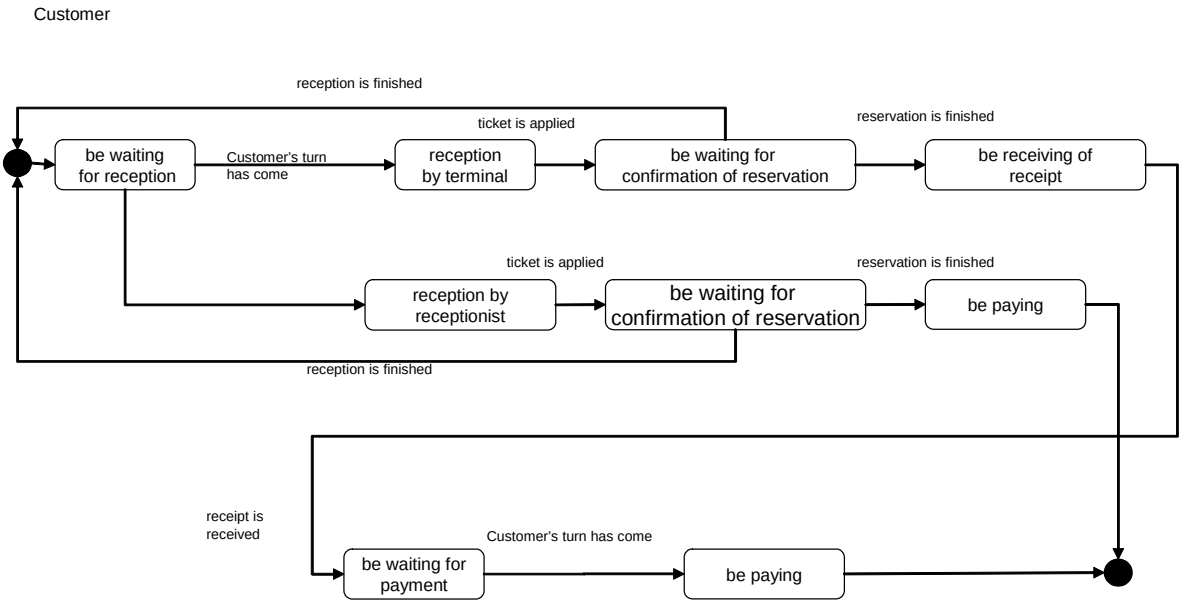

Fig. 10 STD.

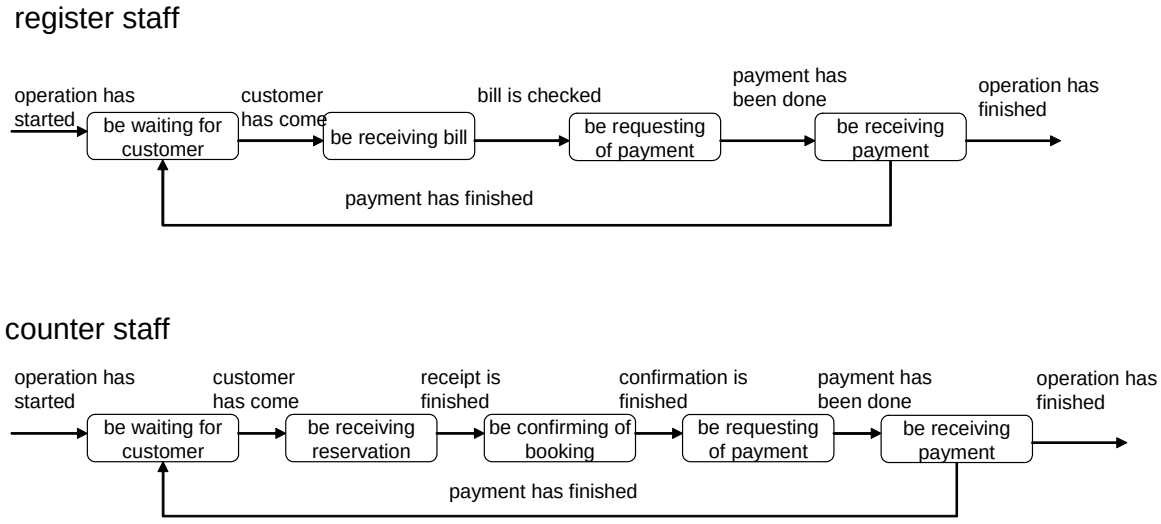

Fig. 11 STD.

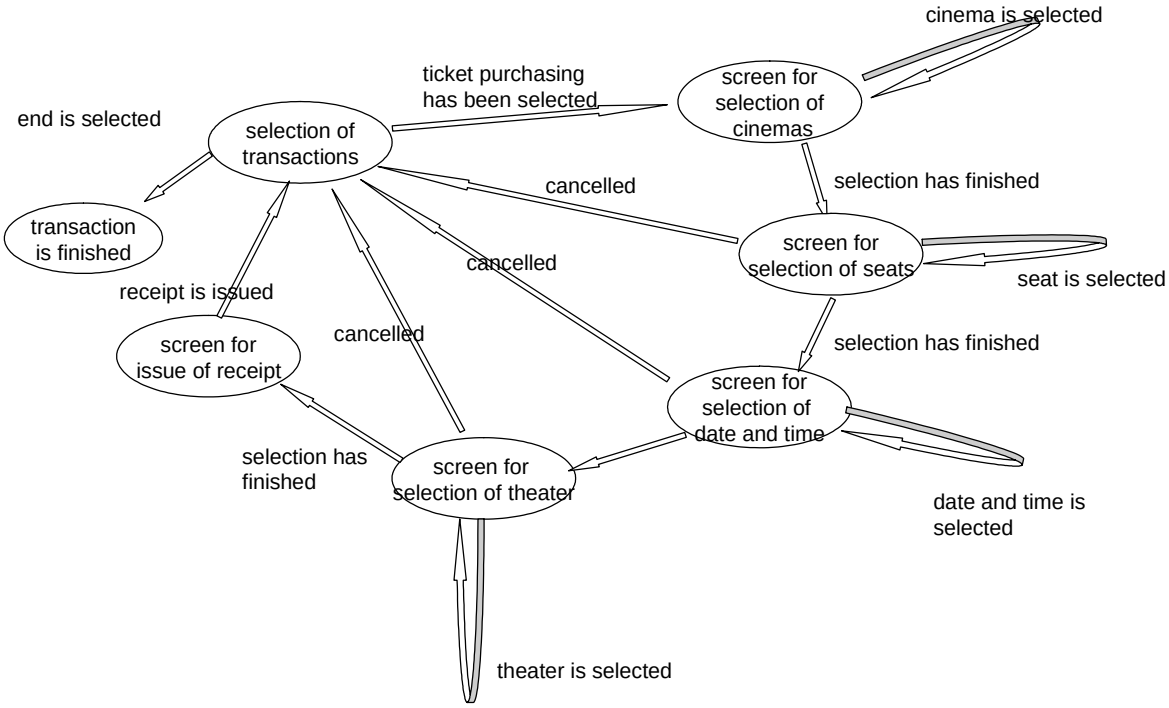

Fig. 12 STD for the Screen.

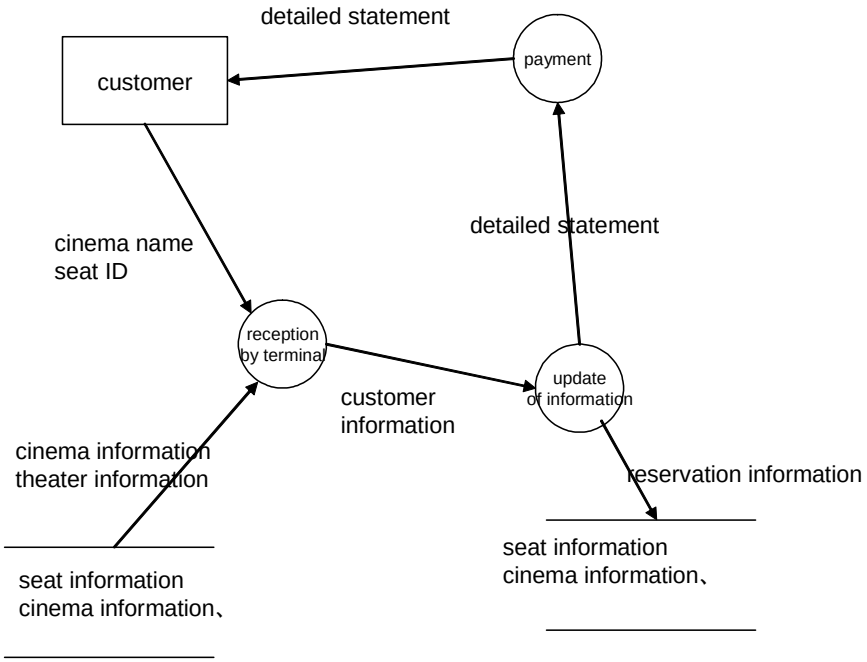

Fig. 13 DFD.

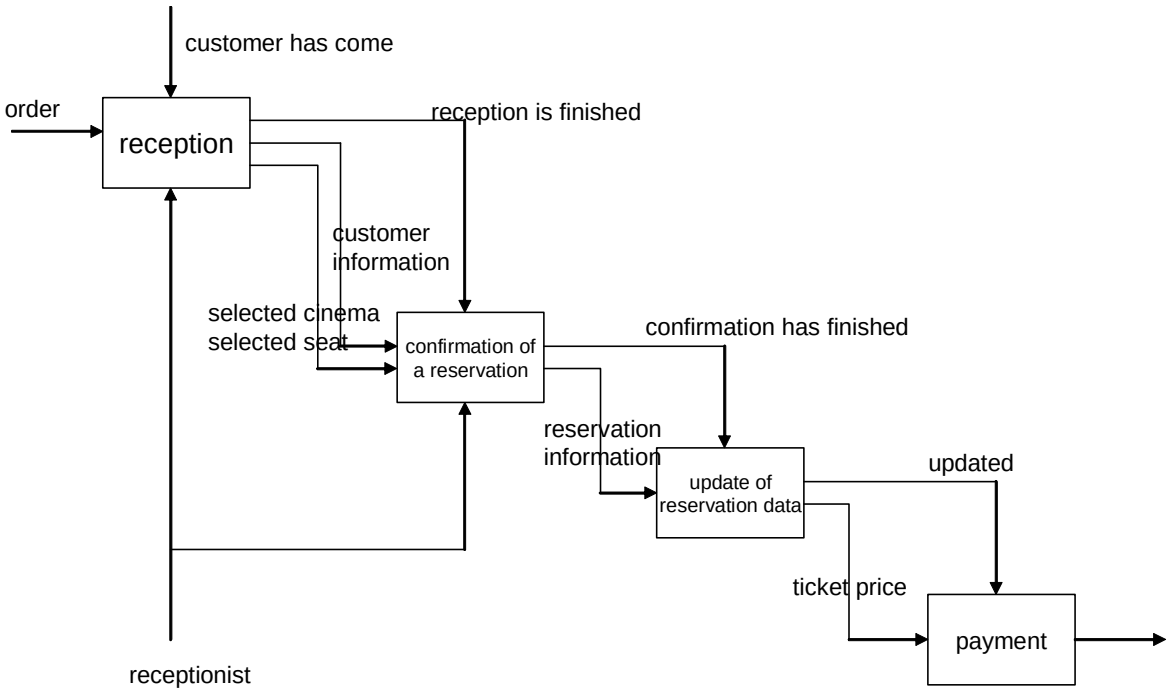

Fig. 14 IDEF0.

10. Concluding Remarks

It is easy for learners to understand the system from various viewpoints by using diagrammatic representation than only text. Our analysis process facilitates them to understand the difference between various viewpoints.

It is effective for increasing the understanding of a diagrammatic analysis to implement the application based on resulted diagrammatic representation. Petri Net Visualizer and Microsoft Access are effective tools to teach Petri Net and E-R Diagram respectively.

In an actual system, various kinds of procedures such as registration of user, processing of lending, processing of payment, etc. are performed. These procedures consist of some routines. Taking a procedure of lending book as an example, the procedure consists of the following routines.

(a) input of a user name
(b) input of a book number
(c) authentication of a user
(d) check whether a user can borrow a book
(e) decrease the number of books which a user can borrow
(f) calculate the date of return

Each of them except (d) can be executed on a standalone basis (which can also be executed by hand) with a database constructed in our class, but they cannot be executed serially. Although it is necessary for understanding the system development to implement the series of routines, time is not enough for teaching programming. Preparing sample program codes which can be executed by some modification is one of the solutions.

11. Acknowledgement

This research has been supported by the fund of Open Research Center Project from MEXT of Japanese Government (2002-2006).

12. References

Kumagai, S., Hirota, T., Kawabata, R., Itoh, K., 2004, "Role-responsibility analysis for cross organizational business process modeling - Multi-disciplinary diagrammatic representations for project management", 2nd International Conference on Project Management (ProMac2004) Proceedings, Chiba, Japan, October 12-14.

Kawabata, R., Itoh, K., Hirota, T., Kumagai, S., 2005, "Effective Diagrammatical Systems Analysis Using The Interrelationships among Use case diagram, Petri net and IDEF0 Diagrams", 2005 Integrated Design and Process Technology Proceedings, Beijing, Chaina, June 13-17.

Peterson, J. L, 1981, Petri Net Theory and Modeling of Systems, Englewood Cliffs, NJ, Prentice Hall.

Kawabata, R., Tabata, S., Itoh, K., 2004, "System Analysis and its Tools for Collaboration Task by Petri Net", 2004 Integrated Design and Process Technology Proceedings, Izmir, Turkey, June 28- July 2.

Senuma, Y., Kawabata, R., Itoh, K., 2003, "Evolutional System Analysis and its Tool by Effective Use of Chart Representations", 2004 Integrated Design and Process Technology Proceedings, Austin, Texas, December 3-6.

Harel, D., Lachover, H., Naamad, A., Pnueli, A., Politi, M., Sherman, R., Shtull-Trauring A., Trakhtenbrot, M., 1990, "STATEMATE : A Working Environment for the Development of Complex Reactive Systems", IEEE Transactions on Software Engineering, Vol.16, No.4, pp. 403-414.

DeMarco, T., 1979, Structured Analysis and System Specification, Prentice-Hall.

Marca, D. A., MacGowan, C. L., 1988, IDEF0/SADT Business Process and Enterprise Modeling, Eclectic Solutions Corp.

Johansson, C., Ohlsson, L., 1995, "A Practival Driven Approach to Software Engineering Education", IEEE Transactions on Education, Vol.38, No. 5, pp 291-295.

Boehm, B., Egyed, A., 1998, "Improving the Life-Cycle Process in Software Engineering Education", European Software Day Proceedings, Västerås, August 27,

Mayer, B., 2001, "Software Engineering in the Academy", IEEE Computer, Vol. 34, pp.28-35.

Bagert, D.J. et al., 1999, "Guidelines for Software Engineering Education,Version 1.0" www.sei.cmu.edu/collaborating/ed/workgroup-ed.html (cited November, 1999).

Bourque, P., Dupuis, R., Abran, A., 1999, "The Guide to the Software Engineering Body of Knowledge", IEEE Software, Vol.16, No. 6, pp.35-44.