ソフトウェア工学 122－7
（1 9 9 9. 3. 1 8）

データストリームダイアグラムの提案とこれを用いた設計手順についての一考察

葛山善基* 神代知範**

あらまし： 本稿では、ビジネスプロセスにおける同一データの流れを分かり易く記述したデータストリームダイアグラム(DSD)を定義する。また、DSD の左右に事象トレースを加えた事象トレース付き DSD (DSDET)と呼ぶ新しいシステム記述モデルを提案する。このモデルはビジネスプロセスの業務分析をする際に、システム全体のデータの流れを従来のデータフロー図（DFD）よりもわかりやすく記述するとともに、流れるデータを単位化しプロセスの実行手順を分かりやすくしたものである。さらに、これらの図を記述するための段階的な手順に関してデマルコの例を用い示した。

A system design method using Data Stream Diagrams with Event Traces

Yoshiki Katsuyama*, Tomonori Kumashiro**
*Faculty of Economics
Shiga University
**Graduate School of Information Science
Nara Institute of Science and Technology

ABSTRACT: In this paper, we define data stream diagram(DSD) as data flow diagrams of the same data for the business process domain. We propose the use of a form of DSD which includes an event trace to the right and left of the DSD (DSDET). Furthermore, we propose an object-oriented system design method. This system design method using the DSDET makes a DSD easy to understand and allows a system analyst to analyze systems easily.

1 INTRODUCTION

DeMarco introduced Data Flow Diagrams (DFD) in (DeMarco, 1978). The DFD has often been used as a model of the business process. This DFD shows a flow of the data between the processes. And also The DFD shows the information that is needed to be stored in a system. The DFD offers important information for a database design. Rumbaugh has admitted the importance of the DFD. But he separates the DFD from the database design. And he showed an importance of an event trace (Rumbaugh, 1991). Jacobson advocated describing the event trace with process (Jacobson, 1992). This diagrams is called sequence diagrams. This description method makes programmer create a program from the event trace easily. But this description skips over the DFD. And Unified Modeling Language(UML) is based on this Jacobson's event trace (Booch, 1997). Therefore, UML does not describe the DFD.
We assert that the DFD is important for the database design. And also we show a several problems of the DFD. There are a few rules to describe the DFD. Even a SPM of DATARUN only divides Company's DFD into several department's DFDs (Pascot 1996). But, it is not enough to catch the stream of data. In order to make the DFD clear, we classified a flow of an object and money and information by coloring data flows of these things. But, when the DFD is described freely, even if the DFD is colored, The DFD doesn't become clear. We thought that it is necessary to arrange the data source and sink at the right and left side of the DFD. If we describe the DFD so, the DFD becomes simple to understand. Furthermore, we thought as follows. If we treat the data exchanges between the process and the data source/sink as events. We can translate the DFD into a new model DFDET that arrange the event trace at the right and left side of the DFD (Katsuyama, 1996 and Katsuyama, 1998). However, the DFD doesn't handle a unit data, so we can not treat the data exchanges in the DFD as events. Therefore we decided to define a new model DSD that handle unit data and the stream of them. Also, we assert a new description method called DSDET that combined this DSD and event trace. The DSDET clarifies that the process exchanging data with the data source and sink, and this DSDET clarifies the execution step of these exchanges. Therefore, a flow of data becomes furthermore evident, when a system is described by the DSDET.
In this paper, the definition of DFD and the event trace are shown in chapter 2. The new model DSD and DSDET are shown in chapter 3. Also, the series of system design from the event trace to DSDET will be showed. First of all, DSD and DSDET of a main object and the information are described. This diagram is clearer than full specification diagrams. Next data flows that support the diagrams are added. The chapter 4 shows the description method of the main DSD. The chapter 5 shows that of the main DSDET. And finally, the chapter 6 shows how to add the support flows of the main DSDET.

2 DEFINITION

2.1 Data Flow Diagrams(DFD)

Data flow diagrams (DFD) use a number of symbols to represent systems. The DFD represents a kind of flow data. Fig. 1 shows the function of system components. Conveyor system gets container from a shipping company and stores them to a warehouse.

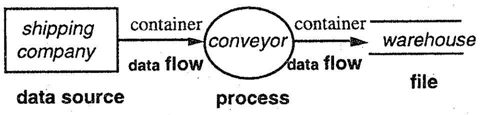

Fig.1 Symbols of DFD

2.2 Event Trace

When an event occurs at an entity, information of the event is sent to other entities. Therefore, when information is sent from one entity to another, we can recognize the event as occurring at the source entity. The data exchange between two entities can be described as follow(Fig.2).

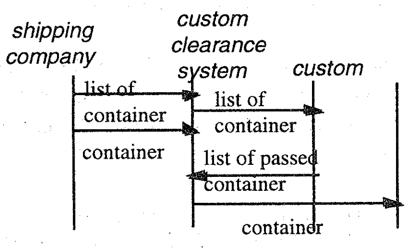

Fig.2 An example of event traces

3 A NEW MODEL FOR THE BUSINESS PROCESS

We propose a new system specification model called Data Stream Diagrams with event trace (DSDET) that combines Data Stream Diagrams (DSD) with event traces. We describe events in DSDET. So, we divide flow data into a unit data, and describe how a unit data flows between two events in DSDET. Furthermore, in order to describe events of a unit data and those of a grouped data together, so loop operation is permitted at event trace diagrams.

3.1 the definition of DSD

If the domain is restricted to business process, and a flow data is divided into a unit data. We can define a process execution like petri nets. So the process executes at the time that the process gets input data. For example, even if two processes that differ in timing were described at one process node in the DFD. We describe these processes as different process nodes in the DSD, so that two independent data flows are distinguished each other (Fig.3). Therefore, DSD makes the data flow and the processes execution clear.

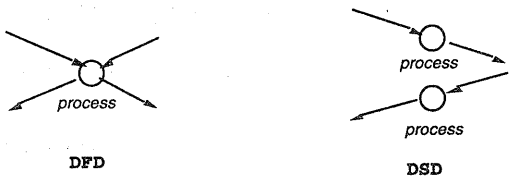

Fig. 3 Execution of processes in DSD

We describe the DSD with the concept of petri-net. But, in our description, we use the same symbols as that of DFD. A symbol of bubble shows a process. A straight line shows synchronization and trigger of the execution. A straight line for synchronization is attached to a front of a process and also that is attached in the process. If a process is implemented after the synchronization, a straight line is attached in front of the process. If the synchronization is taken at the implementation of a process, a straight line is described in the process (Fig.4). If only one data flow enters process, a straight line is not necessary to represent the trigger of the process. Therefore, we do not describe a straight line.

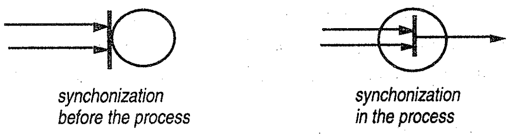
synchonization before the process
synchonization in the process

Fig. 4 Flow control of DSD

Even if two data flows enter the same process node, these flows doesn't always synchronize. There are two types of data flow. One is a push type of data flow that activates a process, Also, the other is pull type of data flow that is pulled by the process. A push type of data flow attaches to the straight line at the front of process. On the other hand, a pull type of data flow attaches to a process directly (Fig.5).

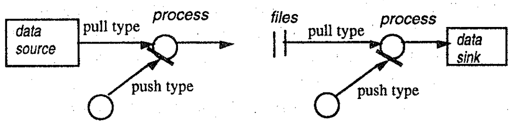

Fig. 5 Two types of data flows in DSD

In the DSD, we distinguish the streams of objects and those of the information. Therefore, first of all we describe a stream of a main object with thick lines. Next we describe information flows with thin lines. A flow of same data is recognized from DSD. Fig. 6 shows a stream of a container with a thick line, and shows a stream of the container information with a thin line. A container flows from a shipping company into an agent. The information of the container controls the flow of the container.

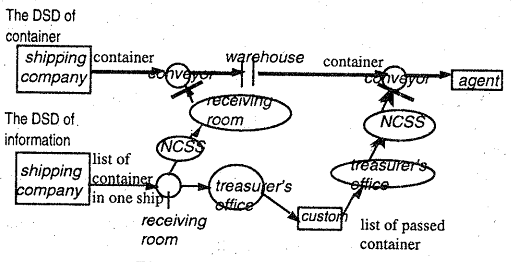

Fig. 6 An example of the DSD

3.3 DSD with event traces (DSDET)

We define a DSD with event trace as one that has an event trace at the left and right of the DSD. This event trace represents the exchange between the processes and the external data source or sink. In a conventional event trace diagram, a horizontal line between two vertical lines that represent the target system and the external data source or sink respectively represents an event. But, in our proposed DSD with an event trace, two vertical lines represent the boundary between the system and the external data source or sinks. Two boundary lines divide three areas, the left, the middle and the right. Sources or sinks are put in the left or right area. All processes and files are put in the middle area. An arrow that crosses the boundaries represents a data exchange between the system and these external data source or sinks (Fig.7). Processes that exchange data with sources or sinks take place after the execution of the preceding data exchange events in the event trace diagrams.

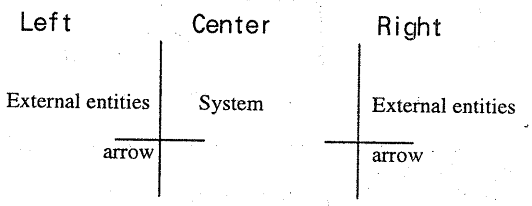

Fig. 7 An example of the DSDET

3.4 grouped data and loop statements

What are unit data and grouped data of flowing data? For example, at customs clearance system, the container is a unit data. The containers in one ship are handled at the same time in the customs. So, the list of container in the ship is a grouped data (Table 1).

Table 1 unit data and grouped data of object

	object	information
unit data	container	
grouped data		list of containers

Loop statements and loop end statements are used at the event trace of DSDET, so that DSDET handles unit data and grouped data at the same diagrams. A loop statement is put before repeated events, while a loop end statement is put after these events.

Fig. 8 is a customs clearance system in DSDET.

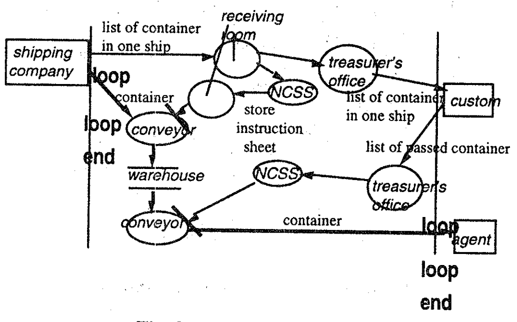

Fig. 8 An example of DSDET

3.5 A characteristic of the DSDET

Event traces and the DSD are described in DSDET. So, it is easy to see the relation between event and process. The DSD is described at the middle part of DSDET and restricted by event trace at right and left side of DSDET. We can not describe a contrary data flow. So, the middle part of DSDET is, what we call, a clean room.

4 AN OBJECT-ORIENTED SYSTEM ANALYSIS METHOD FOR THE BUSINESS PROCESS

This method contains two phases. One is to describe DSD. Second is to transform DSD to DSDET. In this chapter, we explain the first phase. The first phase is divided into five small steps; to describe a system outline, to select objects, and to describe the event trace for objects, and to analyze event trace for the same object, and to describe the DSD for two events of the same object.

4.1 Describing the system outline

The system outline describes what kinds of data are exchanged between the target system and external data source or sinks. In the case of a customs clearance system (Fig. 9), which is a famous example of a DFD in the book(DeMarco, 1978), the system outline is as follows: the system get containers and the list of them from the shipping company, and requests a customs inspection, while the containers are kept in a warehouse. Containers whose inspection is finished by the customs are handed over to an agency (Fig.10).

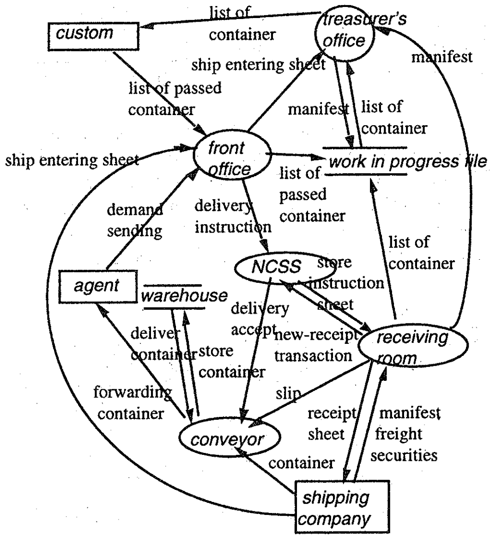

Fig.9 An famous example of DeMarco

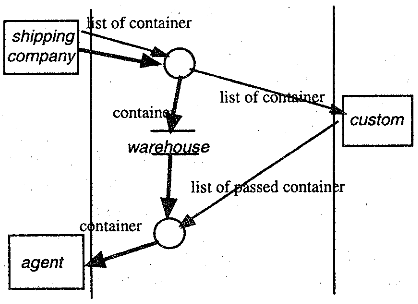

Fig. 10 outline of system

4.2 Finding objects and object information

There are external objects and object information about them, which appear from sources and enter sinks. The objects that flow only in the target system are called internal objects and are distinguished from external objects.

(1) An example of an external object is a container.

(2) An example of external object information is a list of container.

(3) An example of an internal object is information on warehouse vacancy.

4.3 Constructing an event trace diagram

External object and object information are described in the same event trace diagram. Fig. 2 shows an event trace diagram of the custom clearance system.

4.4 Constructing a DSD for each external object and object information

(1) Analyzing the event trace

To construct a DSD for each external object, first of all the event trace is analyzed as follow, and DSD information for each external object is obtained.

(i) Choose a main object from external objects.

(ii) Find unit data or grouped data of the object.

(iii)Select events involving this data of the object from the event trace diagrams.

(iv) Connect these events to describe the stream of this data.

Repeat step (ii) and (iv) for other objects and object information, in order to describe the stream of all the objects and object information.

A cargo is a main object in the custom clearance system, and a container is a grouped data of cargoes. First of all, the event trace for a container has to be analyzed, by selecting the events involving this container, and connecting them together. Thus it becomes clear that the container enters the system from a shipping company, is stored in a warehouse, and is finally sent to an agency.

There is a list of containers, which was sent to the system from a shipping company. This is sent to the customs and then back to the system again.

Finally, the system sends out only containers that have been passed by the customs (Fig. 11).

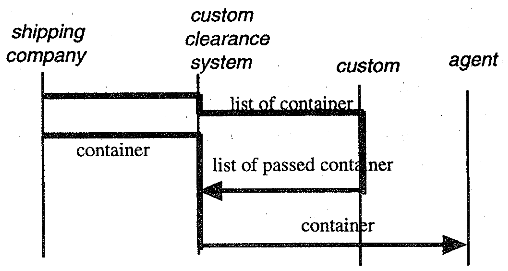

Fig. 11 Connecting events of the same data

(2) Constructing the DSD for all external objects and object information

The DSD for each object will be constructed as follows.

(i) Choose the main object.

(ii) Find unit data or grouped data of object or information.

(iii)Construct data streams by connecting events involving the data.

Repeat the process (i) - (iii) for other objects and objet information. Thus, a DSD is constructed for all objects and object information.

In the case of the custom clearance system, the main object is container. Therefore, first of all, we have to follow events involving that container. Through these

events, the container moves from the shipping company to the agency via the customs clearance system. The DSD for the container can be constructed by tracing the connections between these events.
Furthermore, the information on the container flows from the shipping company to the customs via the system and finally back to the system. A DSD can also be constructed for this information (Fig. 6).
NCSS is a system which manages the information on empty space, and which decides where containers should be stored in a warehouse.

5 CONSTRUCTING A DSD WITH AN EVENT TRACE

The DSD with an event trace is constructed by using DSD. First we arrange source and sinks to the left and right side of DSDET. Next DSD is entered in DSDET. Finally the control statements are described.

5.1 Arranging source and sink

If the following conditions are satisfied, it is desirable that two external data source or sinks is arranged on the same side or on opposite sides of event traces.

(a) If the following condition is satisfied, it is desirable that two external data source or sinks are arranged on the same side.
Condition A: There is a process that consists of two events involving external data source or sinks.
Example: Of the container passed by the customs, only those that are required will be delivered.
In this case, the delivery process requires a list of passed container and a list of requests for delivery (Fig. 12). In order that data streams involving these lists do not cross to the other DSD, the external data source which are the source of these data streams must be arranged on the same side of the DSD.

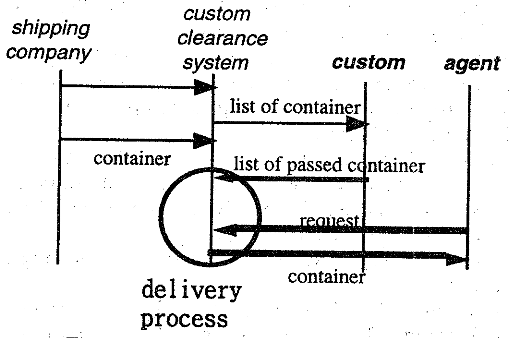

Fig. 12 delivery process consists of two events

(b) In the following condition is satisfied, it is desirable that two external data source or sinks are arranged on opposite sides.
Condition B: Two pairs of events make a circle. And, the pairs of events occur alternately.
Example: The customs clearance system receives a list of containers from a shipping company and the system sends it to the customs. After that, the system moves the container to a warehouse. Inspection of the container is carried out by the customs. A list of containers that pass the inspection is transmitted to the customs clearance system from the customs.
In this case, the following events and related processes are carried out sequentially.
Event(i): transmission of a list of containers from the shipping company to the customs clearance system.
Event(ii): transmission of a list of containers from the customs clearance system to the custom.
Event(iii): delivery of the containers from the shipping company to the customs clearance system.
Event(iv): transmission of a list of passed containers from the customs to the custom clearance system.
Event(i), (iii) came from shipping company. Event(ii), (iv) came to or from customs. Event(i), (ii) are connected by shipping company, and Event(ii) occur after Event(i). Event(iii), (iv) are connected by custom, and Event(iv) occur after Event(iii) because the custom check containers in the warehouse. So, Event pairs(i) (iii) and (ii) (iv) make a circle, and the two pairs of events occur alternately (Fig.13).
It is preferable for the customs to be arranged on the opposite side from the shipping company in the DSD with an event trace, to avoid the DSD crossing
Assuming that the shipping company is arranged on the left side, it is preferable to arrange the customs and the agency on the right side to satisfy these conditions.

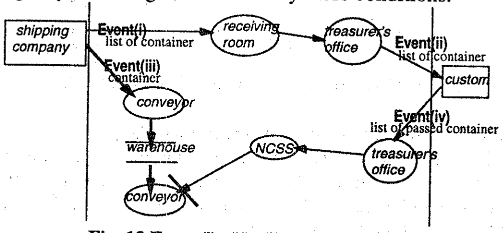

Fig. 13 Event(i), (ii), (iii), (iv) make circle

5.2 Describing the main flow of the DSDET

First of all, the flow of object is entered into the DSD with event trace. Next, the flow of object information is entered in the DSD with event trace. If the exchanges between sources/sinks and the system are continuous, these source/sinks are described as one external entity. The following gives an example of the customs clearance system. First of all, the container moves from a shipping company to an agency. Next, there is a flow of object information. This controls the entry of containers to the warehouse and the exit of containers to agents.

5.3 Describing the control of repetition

When an event of unit data and an event of grouped data are described at the same event trace diagrams, the loop statements are inserted into the event trace diagrams. In the customs clearance system, Fig.8 is obtained.

6 DETAIL DESCRIPTION OF DSDET

We have showed how to describe DSDET of external object. Next, we have to add detail information of accelerating data stream. Finally, we change data exchanges from data source or sinks, so that the data exchanges proceed at only one process called front office. And works in progress are recorded at a file.

6.1 Adding data flows of demand and notification

A list of containers is handed over to customs according to the ships entering the port. The system notifies the shipping company of the places where containers are stored in the warehouse. The containers which have been passed for entry by the customs, and which have been requested by the agency, will be handed over to the agency. These data flows are added to the DSDET(Fig. 14).

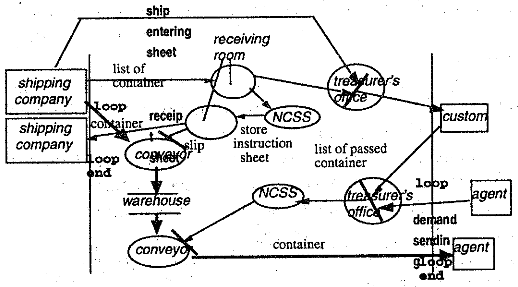

Fig 14 Adding data flows of demand and notification

6.2 Describing the flow of internal objects

In this case, the internal object information is warehouse vacancy information. This information is paired with container information. This data flow is described as a DSD inside the NCSS(Fig. 15).

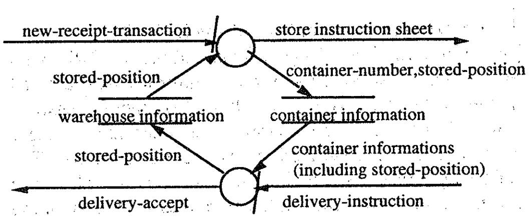

Fig. 15 Data flow of internal object

6.3 Dividing processes and adding a record process

Processes are divided corresponding to an organization. And DSD is changed associated with the office. Several processes leave records to a file after processing. By reading the work in progress file, we can understands which process is going on now (Fig.16).

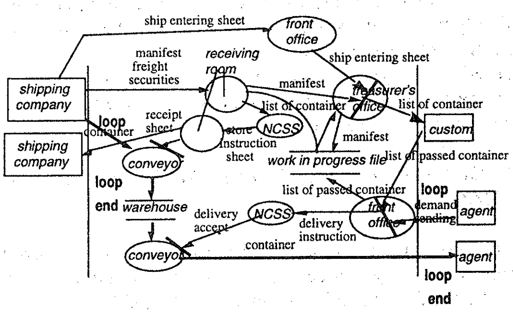

Fig. 16 Dividing processes and adding record processes

7 SUMMARY

We have proposed Data Stream Diagrams(DSD) and DSD with event traces (DSDET), which describes event traces to the right and left of the DSD. The DSD and event traces are described in one diagram, so the system analyst can see the relation between events and processes easily. Furthermore, we have presented an object-oriented system design method that takes into account the business process domain and describes the flows of data with event traces step by step. This proposed type of DSDET makes the relation between DSD and event traces clear. Additionally, the system design method presented with this DSDET gives the system analyst guidance in system design.

REFERENCES:

DeMarco, T., 1978, "Structured Analysis and System Specification", *Yourdon Press*, New York.
Rumbaugh, J., 1991, "Object-Oriented Modeling and Design", *Prentice Hall, Inc.* New Jersey.
Jacobson, I., Christerson, M., Jonsson, P. and Overgaard, G, 1992, "Object-Oriented Software Engineering - A Use Case Driven Approach." Reading, MA: *Addison-Wesley*; New York: ACM Press.
Booch, G., Rumbaugh, J. and Jaobson, I., 1997, "Unified Modeling Language, Version 1.1" Rational Software Corporation, 2800 San Thomas Expressway, Santa Clara, CA 95051-0951 (USA), URL:http//www.rational.com
Pascot, D., 1996, "DATARUN Concepts" *CSA Research Pte. Ltd.*
Katsuyama, Y., Kumashiro, T., 1996, "Structured Specification of Data Flow Diagram" IPSJ 96-SE-111-2 pp9-16
Katsuyama Y., Kumashiro T., 1998, "A system design method using Data Flow Diagrams with Event Traces" IPSJ 98-SE-118-11 pp79-86