Oleh : Kundang K Juman

Data Flow Diagram Tutorial

1. Introduction

This unit deals with one of the major techniques for recording the requirements of a user for a new computer application. An initial diagram is constructed to show the processes which are being implemented in an existing system. The diagram helps to show how information is used to produce the functions that are required by the current system. It also shows what information is provided to the system and what information is provided form the system. Other benefits include the documentation of who is using the system and what data will be stored. By careful construction of the DFDs (data flow diagrams) the boundaries of the system to be built may be clearly identified. This helps to clarify what will and what will not be constructed. It will also show the interaction that may be required with other systems.

The data flow diagrams should also have some associated documentation. This is necessary as the diagrams are meant as a visual representation of the way in which information is processed. There is limited space on the diagrams so that documentation to explain, refine and describe further details of what is shown need to be kept somewhere in the proposed system documentation. The data flow diagrams and the associated documentation together combine to form a data flow model. This is also commonly called a process model.

The user requirements when complete are used as a basis for the development of the system. Later when the system has been developed it can be tested against the initial requirements to see whether the user's needs have been met.

1.1 CASE tools

For many ways of developing and implementing new systems, software is available to help the systems analyst produce what is required. This software is called a CASE tool. CASE stands for Computer Aided Software Engineering. Data Flow Diagrams are usually produced using a CASE tool although they can be produced simply with a pencil and paper. The diagrams shown in this unit were developed using SELECT SSADM Professional version 4.1.1.

Using a CASE tool for construction of the DFDs has many advantages. It is not just a drawing tool. Firstly the CASE tool will not allow a non-standard use of notation for all the items in the diagrams. Secondly it applies some rules so that designing the diagrams prevents the users from making connections between the different items that should not be allowed. There is also some checking that may be invoked to alert the designer to some potential errors. As we will see the CASE tool helps to check that the DFDs are consistent with other views of the system which require diagrams that may also be drawn with the CASE tool.
You would find it helpful to become familiar with a CASE tool to practice some of the example activities in this unit.

1.2 Development and purpose of DFDs

Before attempting to construct an initial DFD it is necessary to gather and digest information that helps us to understand how data is processed in the current system. Fact-finding techniques are used for this purpose and are discussed in another Unit. As the DFD is constructed a systems analyst will often come across areas of doubt where the precise way to model the system is unclear. This is a natural part of the development and should not be regarded with alarm. In fact, it is expected and it is a consequence of attempting to model the current situation that questions will be asked to clarify the exact processes which are taking place. Sometimes the analyst will make an assumption and then check this with the user at a subsequent meeting.

Results of interviews, documents, reports, questionnaires etc. will all play a part in helping the analyst to gain an insight into the current processes. Where a system is being developed form scratch the analyst will work with the user to develop the proposed DFDs.
When all the information about the current system is gather it should be possible to construct the DFDs to show:

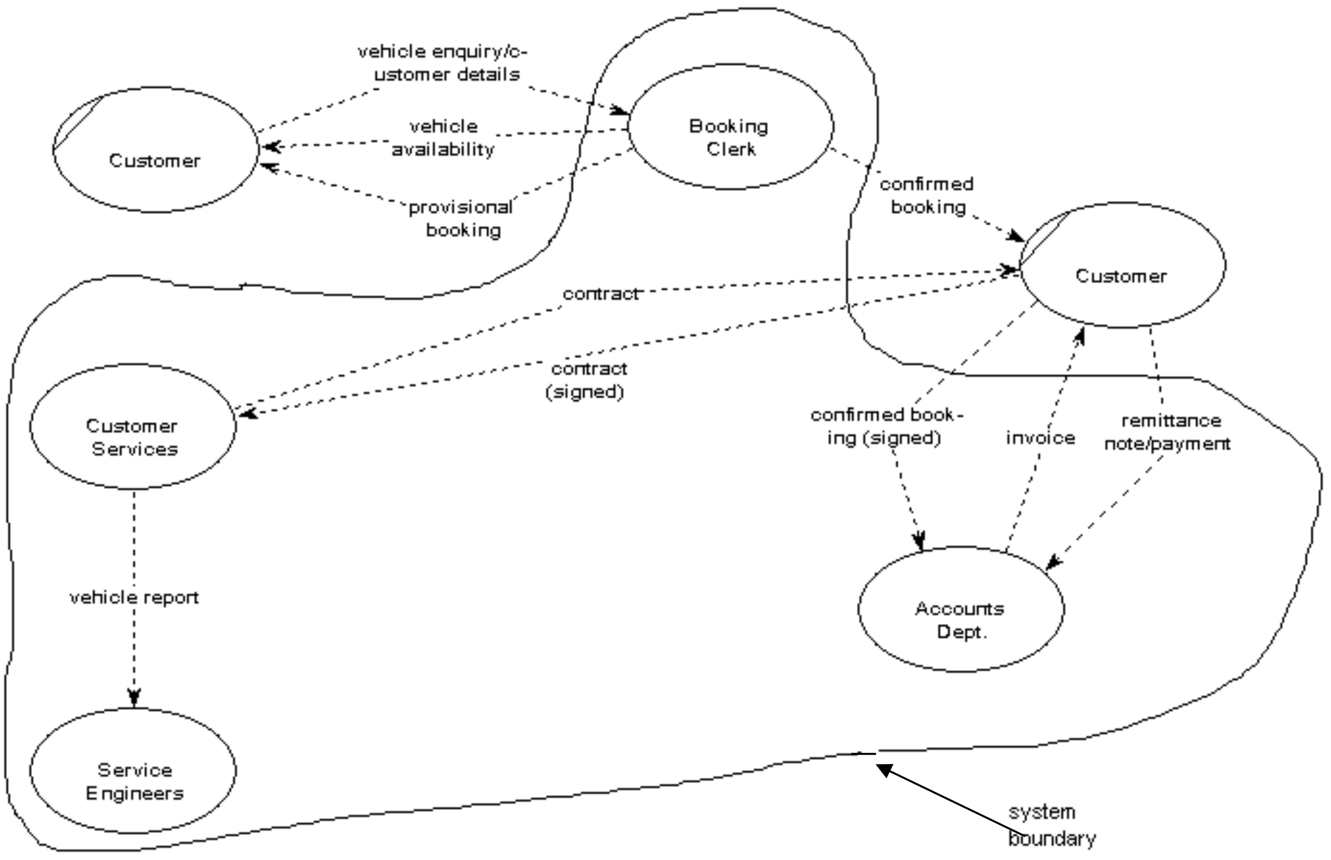
- the information that enters and leaves the system
- the people/roles/other systems who generate and/or receive that information
- the processes that occur in the system to manipulate the information
- the information that is stored in the system
- the boundary of the system indicating what is (and what is not) included

As a staring point, it is sometimes useful to construct a document flow diagram. This diagram shows how ‘documents’ are passed around an organisation to fulfil the current requirements of the system under investigation. The term’ documents’ is interpreted very loosely and usually translates to information.
Sometimes before beginning to produce the set of data flow diagrams a document flow diagram is generated. This helps to establish the system boundary so we can decide which parts of the system we are modelling and which parts we are not. The document flow diagram shows the different documents (or information sources in the system and how they flow from a source to a recipient.

Here is an example of a document flow diagram shown below.

Here ellipses represent either the source or recipient of the 'documents' and named arrows show the direction of transfer and the nature of the information being exchanged. This kind of diagram is a first step in understanding what information is in the system and how it is used to perform the required functions.

Another useful feature of the construction of this type of diagram is that it enables a sensible discussion of where the system boundary should lie. In other words it is important to establish what is to be included in the proposed information system and what is not. To indicate this, a system boundary line is constructed on the document flow diagram.



The DFDs are used to:

- discuss with the user a diagrammatic interpretation of the processes in the system and clarify what is currently being performed
- determine what the new system should be able to do and what information is required for each different process that should be carried out
- check that the completed system conforms to its intended design

2. Components of Data Flow Diagrams

The components of a Data flow Diagram are always the same but there are different diagrammatic notations used. The notation used here is one adopted by a methodology known as SSADM (Structured Systems Analysis and Design Methods)

2.1 Components

Luckily there are only four different symbols that are normally used on a DFD. The elements represented are:

- External entities
- Processes
- Data stores
- Data flows

External entities

External entities are those things that are identified as needing to interact with the system under consideration. The external entities either input information to the system, output information from the system or both. Typically they may represent job titles or other systems that interact with the system to be built. Some examples are given below in Figure 1. Notice that the SSADM symbol is an ellipse. If the same external entity is shown more than once on a diagram (for clarity) a diagonal line indicates this.

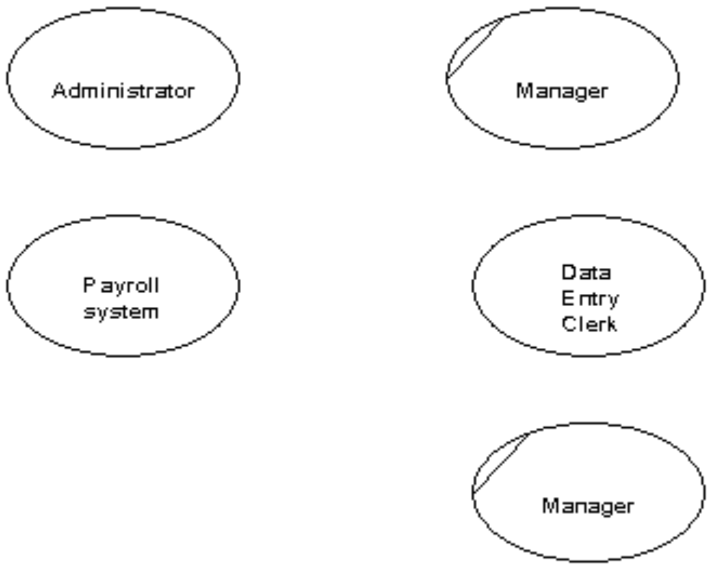


Figure 1 Examples of external entities

Processes

Processes are actions that are carried out with the data that flows around the system. A process accepts input data needed for the process to be carried out and produces data that it passes on to another part of the DFD. The processes that are identified on a design DFD will be provided in the final artefact. They may be provided for using special screens for input and output or by the provision of specific buttons or menu items. Each identifiable process must have a well chosen process name that describes what the process will do with the information it uses and the output it will produce. Process names must be well chosen to give a precise meaning to the action to be taken. It is good practice to always start with a strong verb and to follow with not more than four or five words.

Examples of good process names would be :

- Enter customer details
- Register new students
- Validate sales orders.

Try to avoid using the verb 'process', otherwise it is easy to use this for every process. We already know from the symbol it is a process so this does not help us to understand what kind of a process we are looking at.

The process symbol has three parts as shown in Figure 2.

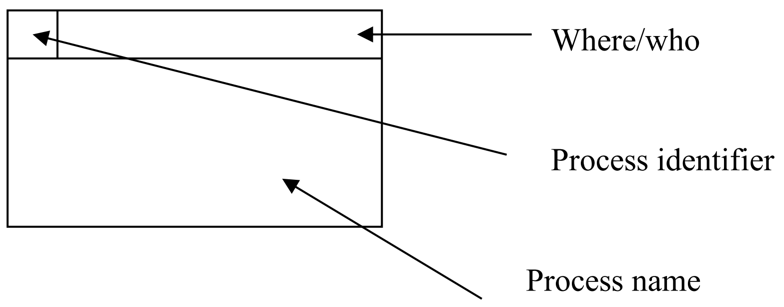


Figure 2 Process Box

The process identifier is allocated so that each process may be referred to uniquely.

The sequence of the process identifiers is usually unimportant but they are frequently to be seen as 1., 2., 3., etc. The top right hand section of the box is used to describe where the process takes place or who is doing the process.

Examples of process boxes are given in Figure 3.

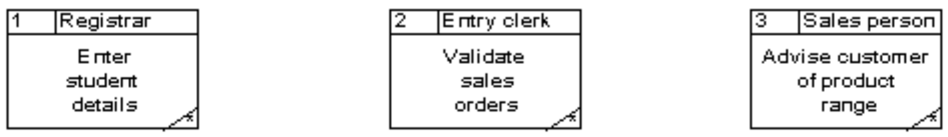


Figure 3 Examples of process boxes

The significance of the asterisk in the bottom right hand corner is discussed in section 3.3, Lower levels of Data Flow Diagrams.

Data stores

Data stores are places where data may be stored. This information may be stored either temporarily or permanently by the user. In any system you will probably need to make some assumptions about which relevant data stores to include. How many data stores you place on a DFD somewhat depends on the case study and how far you go in being specific about the information stored in them. It is important to remember that unless we store information coming into our system it will be lost.

The symbol for a data store is shown in Figure 4 and examples are given in Figure 5.

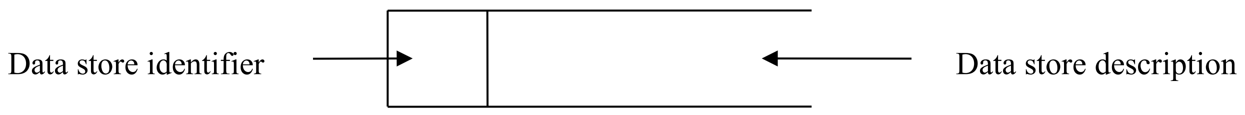


Figure 4 Symbol for a data store

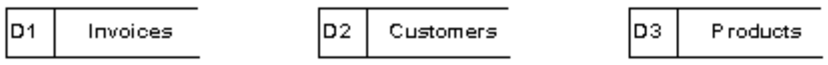


Figure 5 Examples of possible data stores

As data stores represent a person, place or thing they are named with a noun. Each data store is given a unique identifier D1, D2 D3 etc.

Data flows

The previous three symbols may be interconnected with data flows. These represent the flow of data to or from a process. The symbol is an arrow and next to it a brief description of the data that is represented. There are some interconnections, though, that are not allowed.

These are:

- Between a data store and another data store
    - This would imply that one data store could independently decide to send some of information to another data store. In practice this must involve a process.
- Between an external entity and a data store

- This would mean that an external entity could read or write to the data stores having direct access. Again in practice this must involve a process.

Also, it is unusual to show interconnections between external entities. We are not normally concerned with information exchanges between two external entities as they are outside our system and therefore of less interest to us. Figure 6 shows some examples of data flows.

Applicant's name
Customer details
Employee record
Payment

Figure 6 Examples of data flows

2.2 Hints on drawing

Let's look at a DFD and see how the features that have just been described may be used. Figure 7 shows an example DFD.

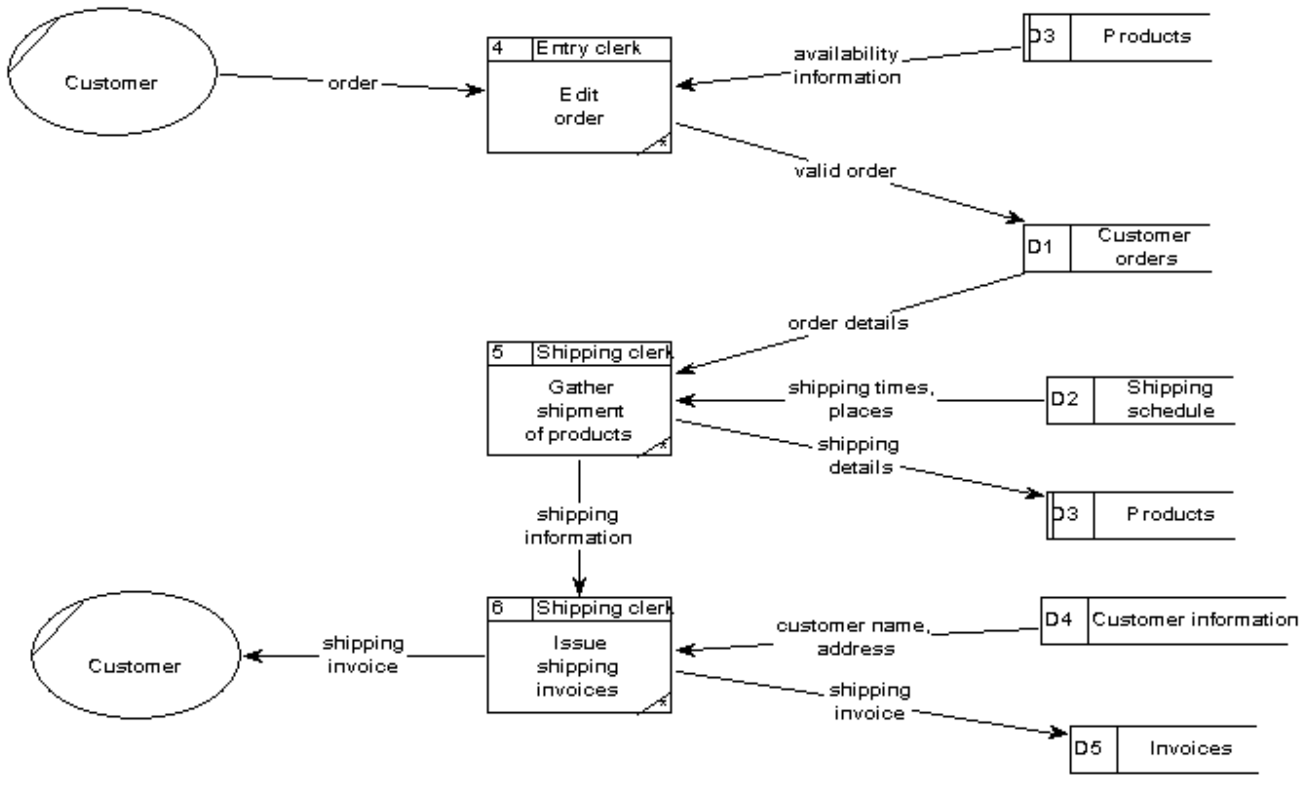


Figure 7 An example DFD

Here are some key points that apply to all DFDs.

- All the data flows are labelled and describe the information that is being carried.
- It tends to make the diagram easier to read if the processes are kept to the middle, the external entities to the left and the data stores appear on the right hand side of the diagram.
- The process names start with a strong verb
- Each process has access to all the information it needs. In the example above, process 4 is required to check orders. Although the case study has not been given, it is reasonable to suppose that the process is looking at a customer's order and checking that any order items correspond to ones that the company sell. In order to do this the process is reading data from the product data store.
- Each process should have an output. If there is no output then there is no point in having that process. A corollary of this is that there must be at least one input to a process as it cannot produce data but can only convert it form one form to another.
- Data stores should have at least one data flow reading from them and one data flow writing to them. If the data is never accessed there is a question as to whether it should be stored. In addition, there must be some way of accumulating data in the data store in the first place so it is unlikely there will be no writing to the data store.
- Data may flow from
    - External entity to process and vice-versa
    - Process to process
    - Process to data store and vice-versa
- No logical order is implied by the choice of id for the process. In the example process id's start at 4. There is no significance to this.

  Drawing diagrams like this requires practice. You should not be unduly worried if your diagram does not look exactly like a colleague's or a model answer. Systems analysis is very rarely done in isolation and you and your working partners and the user will come to some agreement about the final product.

3. Developing Data Flow Diagrams

Data flow diagrams usually occur in sets. The set consist of different levels. To start with a context diagram is drawn. This shows the people and/or systems that interact with the system under construction. By interaction we mean putting information in or taking information out of our system. This diagram gives an overview of the information going in and coming out of the system.
The system is represented by a box. In this DFD (level 0 DFD) no processes or data stores are shown. Another diagram (level 1 DFD) is now developed which shows the processes taking place to convert the inputs shown in the context diagram to the outputs. In this DFD detail is given to show which processes are responsible for accepting the different inputs and producing the different outputs. Any process shown that is complicated by a number of data flows and requires further refinement is shown on another diagram (level 2 DFD) where sub-processes are shown together with a [illegible] a stores. Figure 8 shows how these levels of detail [illegible] one another.

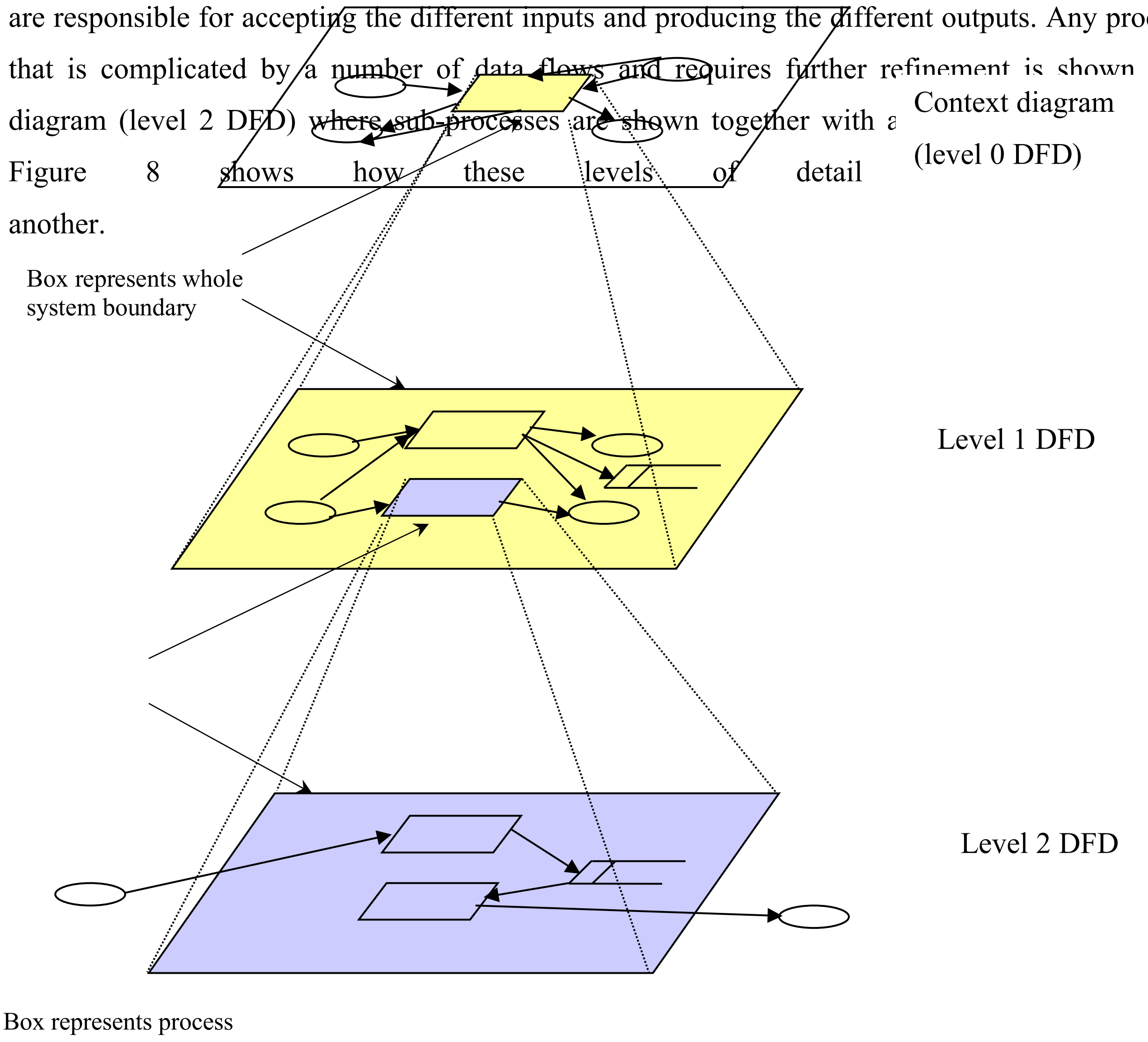
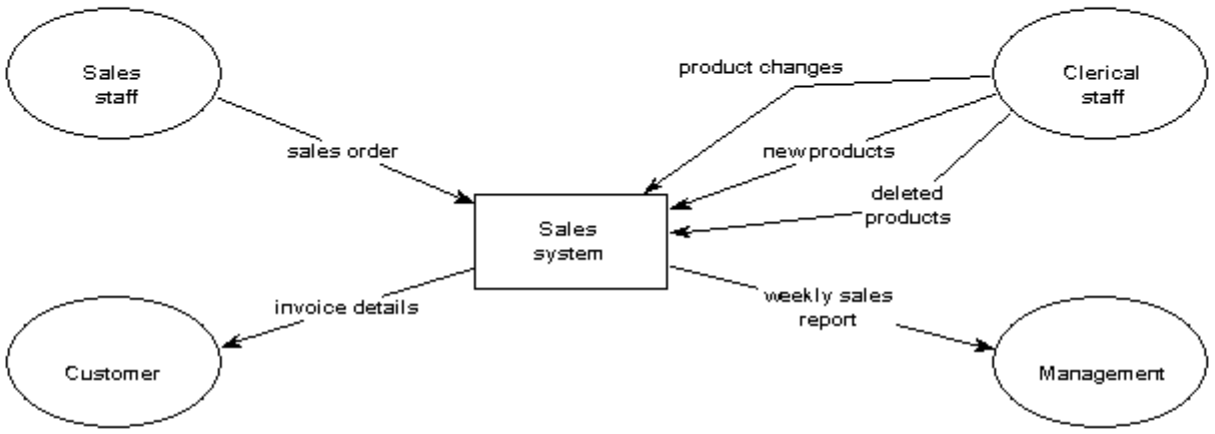


Figure 8 Different levels of DFD

3.1 Context diagram

This DFD provides an overview of the data entering and leaving the system. It also shows the entities that are providing or receiving that data. These correspond usually to the people that are using the system we



will develop. The context diagram helps to define our system boundary to show what is included in, and what is excluded from, our system.

The diagram consists of a rectangle representing the system boundary, the external entities interacting with the system and the data which flows into and out of the system. Figure 9 gives an example of a context diagram.

Figure 9 An example context diagram

3.2 Level 1 Data Flow Diagram

Now we wish to develop further a model of what the system will do with the information the external entities will supply to it. We construct a diagram which is, in effect, taking a magnifying glass to the system boundary represented by the rectangle in the context diagram. We will be looking inside the rectangle and describing what is done with the data inputs in order to provide the data outputs required. The level 1 DFD we construct is a 'child diagram' of the context diagram. We should see all the inputs outputs and external entities that are present in the context diagram. This time we shall include processes and data stores. When students come to construct these level 1 DFDs for the first time they are often unsure as to how many processes to show. It somewhat depends on the case study. However, as a guide it is probable that you would not want more than eight or nine processes as more than this would make the diagram too cluttered. In addition the process names should be chosen so that there are at least three. Fewer than this would mean that the system was unrealistically simple.

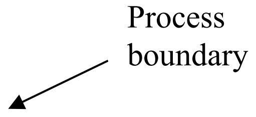
Figure 10 shows a level 1 DFD corresponding to the context diagram of Figure 9.

Note the following features:

- Every data flow on the context diagram, to or from an external entity, is also shown on the Level 1 DFD.
- Each process has a good strong verb describing what the process is doing with the information received.
- Some data stores appear more than once. This is indicated by a double vertical line on the left hand side of the symbol.
- Each process has access to the relevant information to be able to produce the required output
- Each process has at least one data flow input and one output.
- Information input to the system is always stored somewhere and so never lost
- At this stage the invoice data store has an input data flow but no output. This would indicate that the invoice data is never used. Later on in the development of this system we might define a process for checking the invoices have been paid and so this information in the invoice data store would then be used.
- One of the processes (process 8) has no asterisk in the bottom right hand corner although all the others do. This is explained in the next section.

3.3 Lower levels of Data Flow Diagrams

Process 8 in Figure 10 is chosen to be investigated in more detail. A level 2 DFD is constructed of this process in Figure 11.



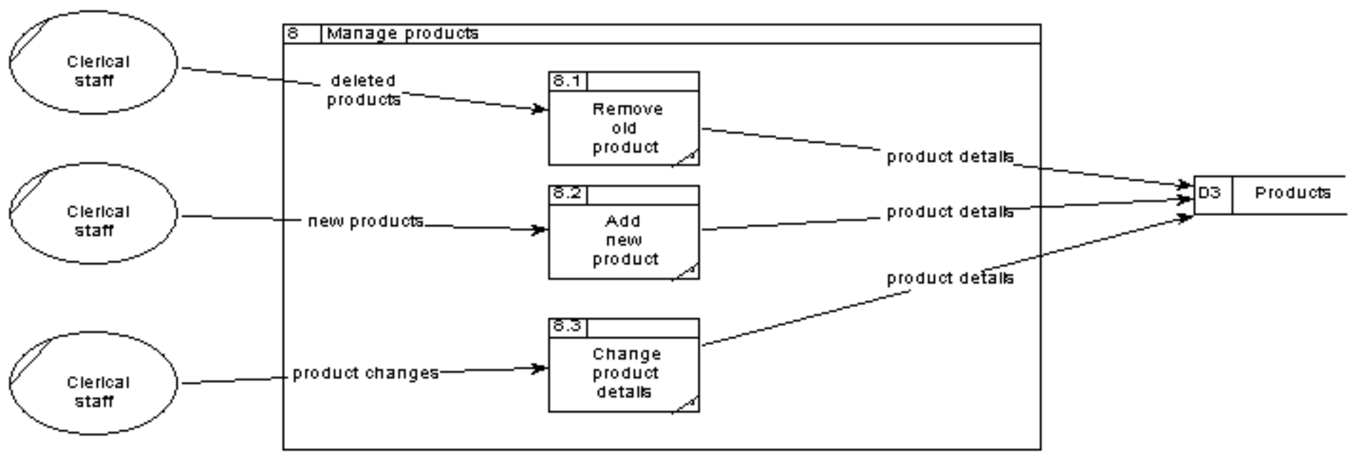


Figure 11 A Level 2 DFD corresponding to Process 8 from the level 1 DFD in Figure 10

The diagram above is a child diagram of process 8 shown in Figure 10. Whenever a process is shown at the lowest level of detail it is called an elementary process. In SSADM this is indicated by an asterisk in the bottom right hand corner of the process box. As process 8 on Figure 10 has a child diagram, the asterisk has been removed by the CASE tool.

You should note:

- All the data flows into and out of process 8 on the level 1DFD also appear on the level 2 DFD
- The numbering of the processes shown on the level 2 DFD 8.1, 8.2 etc. indicates that they are sub-processes of process 8.
- The name of the process, 'Manage products', is shown in the level 2 process boundary title.

Only those processes that merit being expanded need to have level 2 DFDs associated with them. It is also possible to have lower levels of DFD dependent on how complex a particular process is. Although this example hasn't had need for it, it sometimes happens that new data stores may be developed in the level2 DFD. Also new data flows may be shown between the elements inside the process boundary.

3.4 Check list

There are many errors that may occur when drawing data flow diagrams. Here is a check list to help you avoid some of the major difficulties.

- External entities must be people or systems that send information to or accept information form the system to be engineered
- Data flows must always be labelled with the data they contain. Do not put verbs in the data flow description as this implies a process

- Parent and child diagrams should be consistent. Do not show a data flow coming from or to an external entity on a level 1 DFD that isn't shown on the context diagram (and vice versa).
- Check the direction of data flows to and from data stores
- Make sure each process has at least one input and one output
- Each data store should have at least one input and one output on the DFDs somewhere
- Each process name should start with a verb
- Where a process has only two data flows (one input and one output) then check it. Usually a data flow has been omitted.

4. Categories of Data Flow Diagrams

Data flow diagrams may be categorised as either physical or logical. A logical diagram shows how the business operates but not how the system will be constructed. The logical diagram omits details of how the more physical aspects are implemented. The logical model is more to do with the business whilst the physical model shows the whole system. The physical model deals with the way that information is stored, the manual procedures used, the necessity for temporary data collections etc.
One approach adopted in SSADM is to have a number of versions of the DFDs. To begin with, a current physical DFD is developed showing exactly how the current system processes and stores data. The physical parts of this are then stripped out to produce a current logical DFD. From this the limitations and difficulties may be analysed. A clear understanding of the present system evolves which forms the basis for building the new system. The current physical DFD is examined and processes that are unnecessary are removed and new features added. It is hoped by this means to include the essential parts of the old system in the new system. The required physical DFD is thus produced and may then be used to create a required physical DFD.

The overall transition of DFDs is shown in Figure 13.

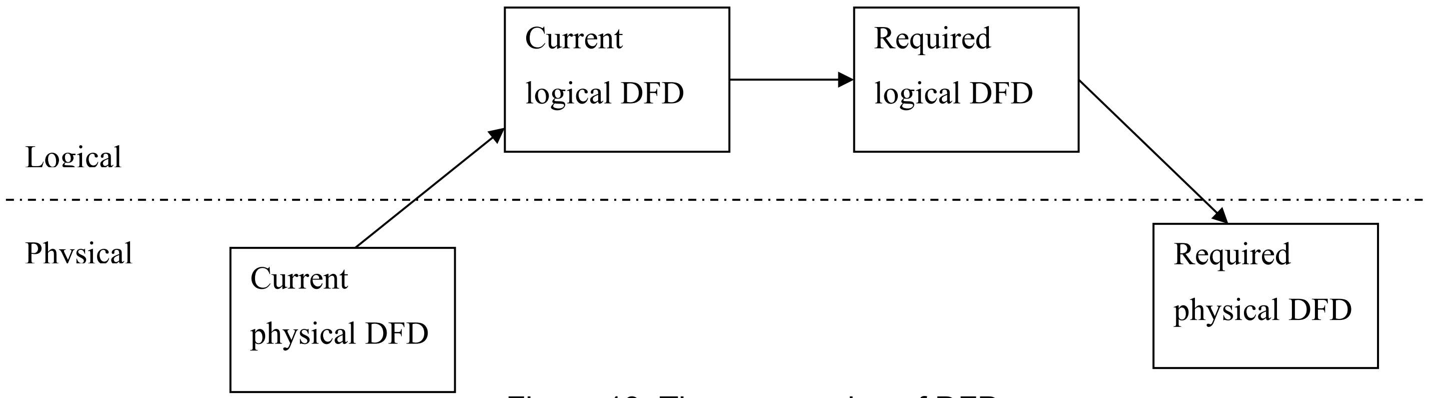


Figure 13 The progression of DFDs

In practice, the first one/two stages are often omitted and the DFDs begin with a construction of the required logical DFD. The logical DFD always appear to be simpler and usually show fewer processes than the corresponding physical DFD.
We will not describe the construction of physical DFDs here but you may like to look at some examples in Kendall and Kendall (1998) or Goodland and Slater (1995)

5. An example of the development of a Data Flow Diagram

As an example of the foregoing discussion we shall use an example to illustrate the techniques described. In order to show the development of a logical data flow diagram we first need to have gathered together information as a result of fact finding. The techniques of fact finding include:

- Observation
  - Observing an individual in the environment in which they normally operate
- Interviews
  - Having a succession of interviews with representatives of the users
- Questionnaires
  - Gaining an overview of the required system and problem areas to be resolved

Having gathered together information relating to the system to be built we may represent it in a textual description, as given below.

Example Case Study – Pizza Supreme

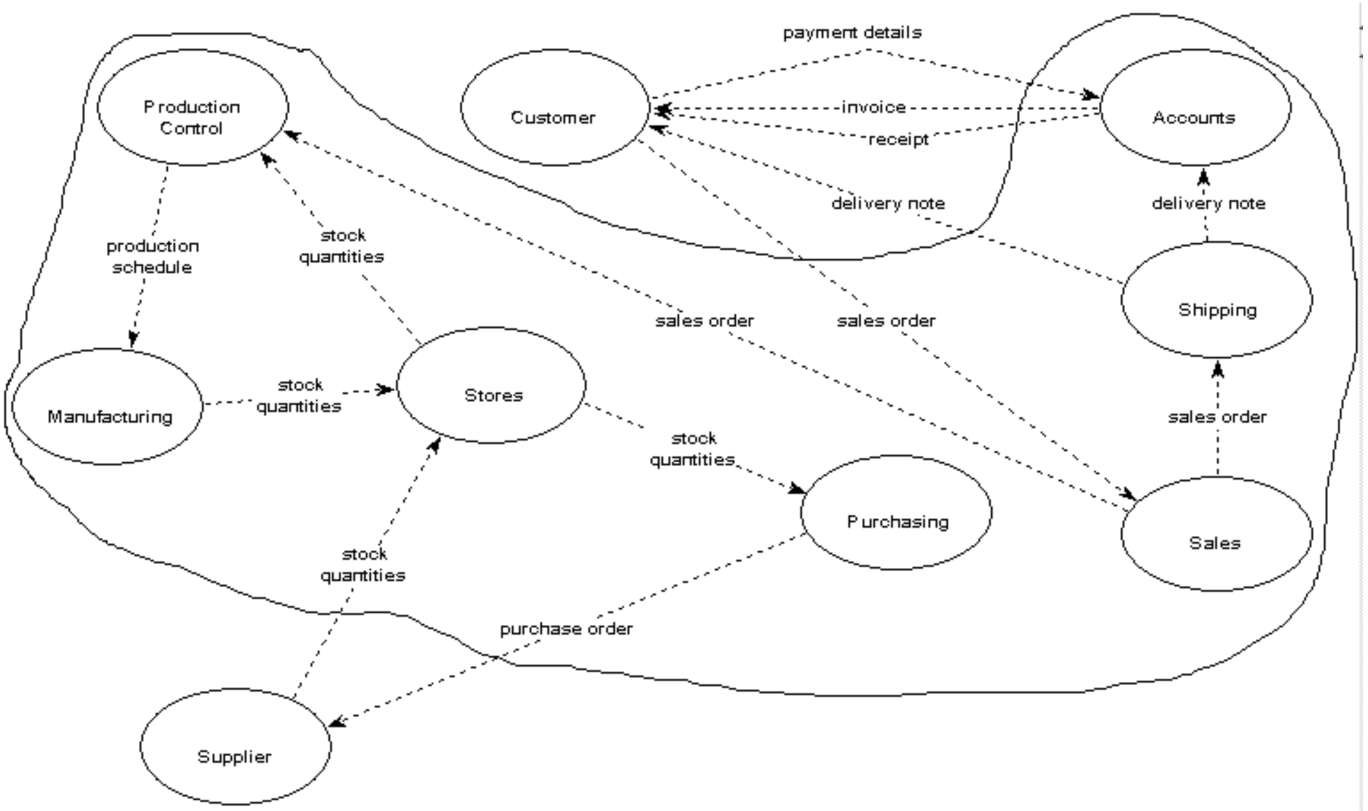
A large pizza business makes pizzas and sells them. The pizzas are manufactured and kept in cold storage for not more than two weeks.
The business is split into a number of functional units. There is Production Control, Manufacturing, Stores, Accounts, Sales, Shipping and Purchasing. Production Control are responsible for organising which pizzas to produce in what order and in what quantity. They need to schedule the production of the pizzas according to the current and expected sales orders together with the number of pizzas already in

Stores. Manufacturing take the raw materials from the Stores and manufacture pizzas returning the completed goods to the Stores. Accounts deal with the payments for the pizzas when delivered to the customer and the payment to the suppliers of the raw materials. Sales deal with customer orders whilst Purchasing organise the buying of raw material from suppliers. Shipping manage the packing and delivery of the goods to the customer with a delivery note.

When a sales order is received by sales they record what is being ordered and by whom. They also record the details of the expected date of delivery. Production Control access this information and make sure that, if required, pizzas are produced by Manufacturing and are ready in Stores for when the delivery needs to be made.

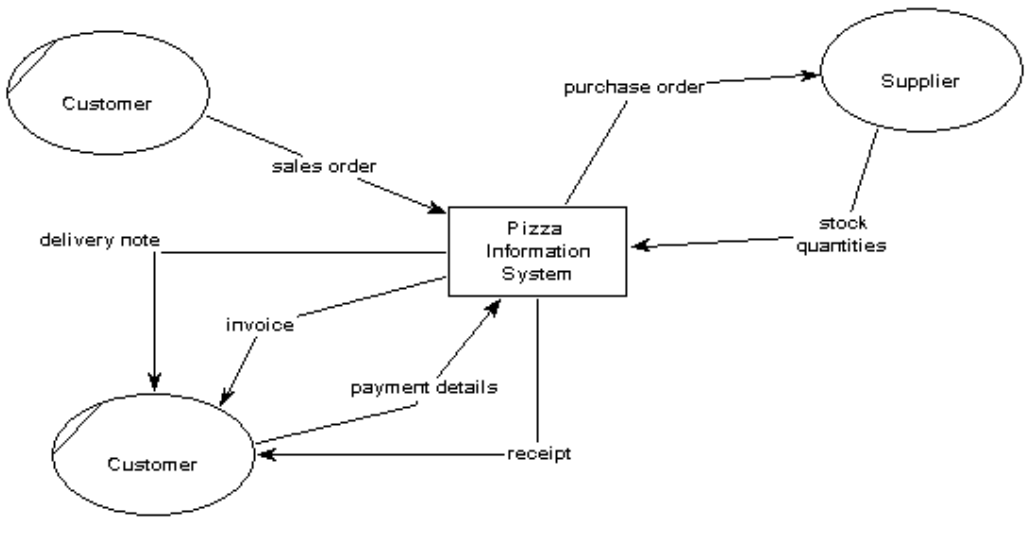
After the delivery is made Accounts make sure that the customer receives an invoice and that payment for the invoice is received at which time a receipt is issued. Purchasing look at the current stock of raw materials and by using current stock levels, supplier turn around times and quantity to be ordered decide what needs to be ordered on a daily basis. Their aim is never to run out of an ingredient but to minimise the amount of raw material kept in stock.

Sometimes a document flow diagram may be drawn to show the way in which information flows around the system. Here is one shown below.



Some of the details here have been assumed and would need to be checked with the user to make sure they are a true reflection of what is happening. By producing this diagram the analyst is gaining an understanding of how the current system is working in terms of the information that is being used and how it is being passed around the system. Here we assume the system to be produced will include all the functions provided by the different functional areas but the suppliers and customers are regarded as being outside the system.

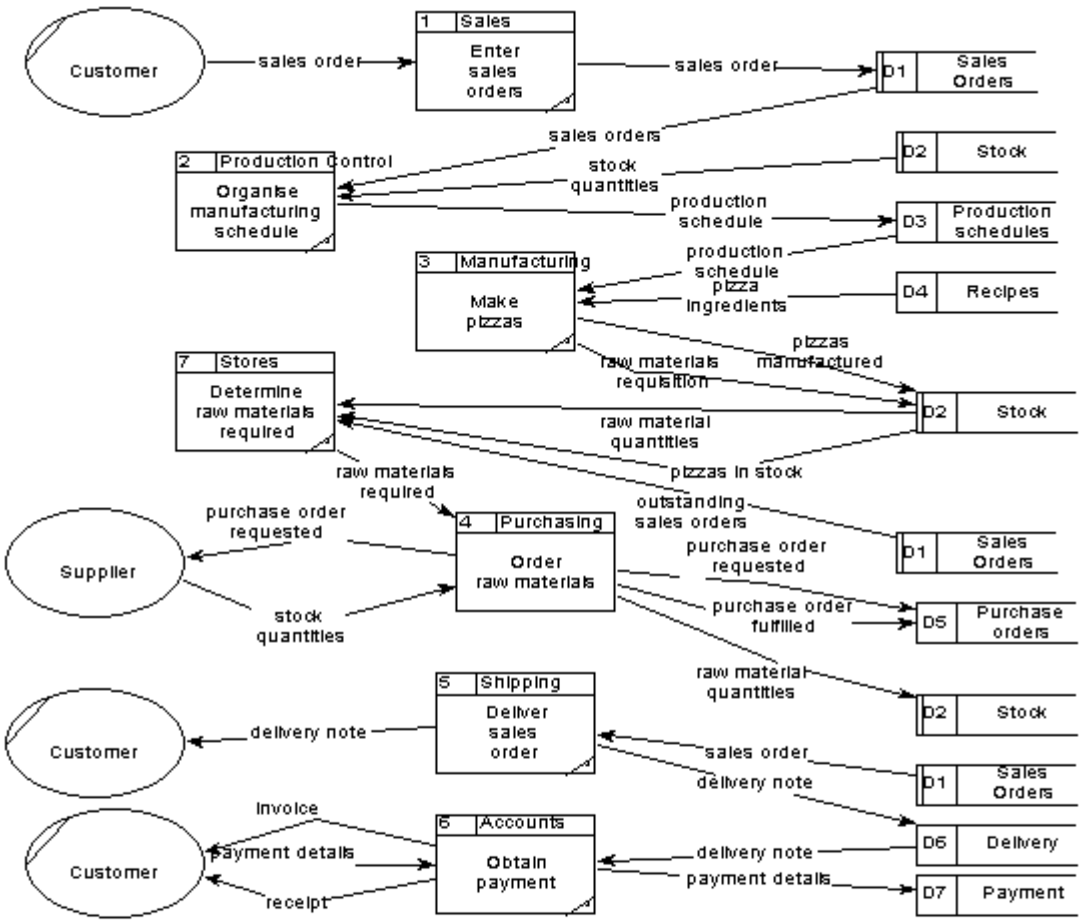
Form this document flow diagram we may construct a context diagram. This will show the information entering and leaving the system. It also shows which external entities are supplying or receiving the



information.

Now try Exercise 1 (in Section 8 below)

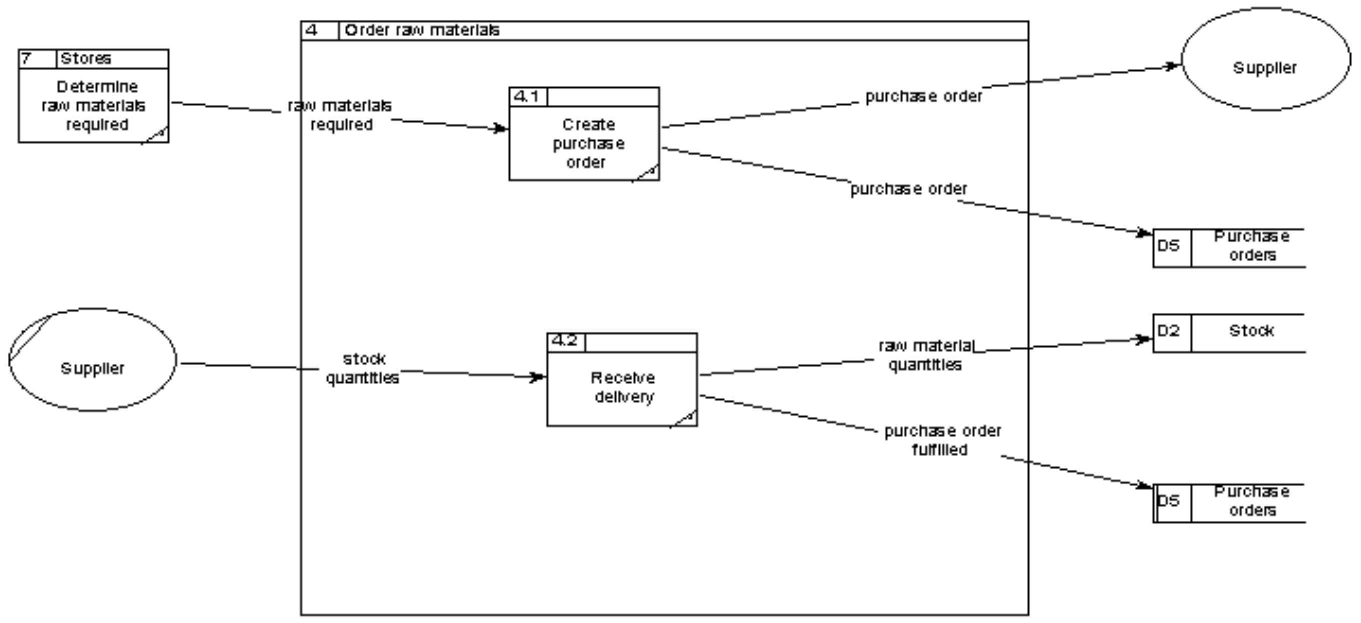
We may now develop a level 1 data flow diagram from the context diagram. The level 1 data flow diagram must show the same data flows form and to the external entities as appear on the context diagram. If this is not the case there would be an inconsistency. If a CASE tool is used this will help to maintain that consistency. We need to decide which processes are handling the information input to the system and which are producing the information outputs that are shown on the context diagram. We also need to think about the process names involved and which data stores are required. Remember if data is not stored away it will be lost !



Now try Activities 2, 3 and 4

Looking at this level 1diagram we see that process 4 'Order raw materials' has a number of incoming and outgoing data flows and to explain what is happening probably needs some decomposition. This then is a candidate for a level 2 DFD. Before looking at the level 2 DFD for this process examine the level 1 DFD and make sure you understand why all the data flows are as they are. Check the diagram against the check list given in section 3.4. You should find this raises at least one problem that will need to be resolved !

Check the level DFD is consistent with the level 1 DFD (its parent diagram).

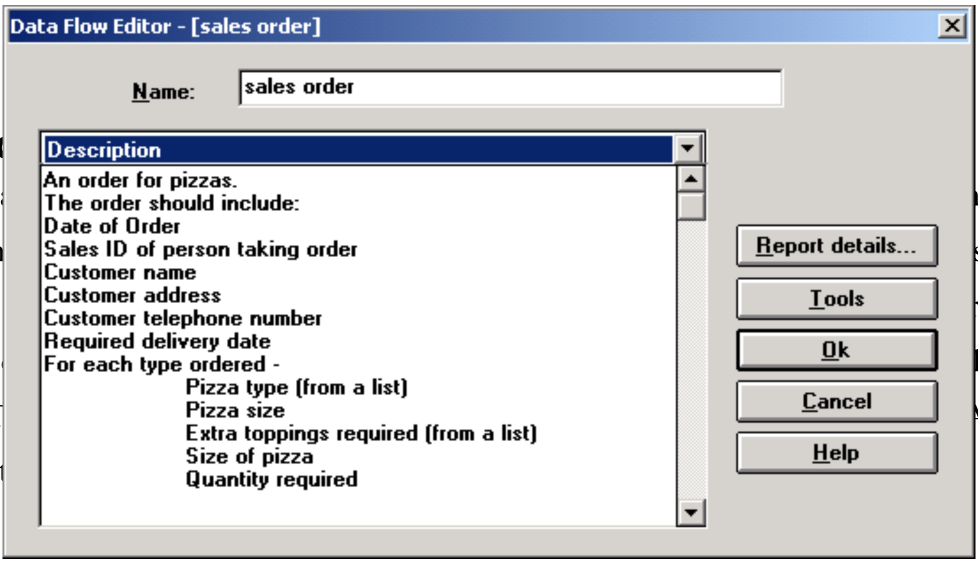


This completes the example showing how Data Flow Diagrams may be constructed. We have produced an outline of the processes that are currently occurring in the physical system. We could now continue to generate a set of current logical DFD's and from there build a set of required logical DFD's. Of course the required processes differ from the current processes in so much as when a new system is produced it should try to solve any of the current problems with the old system and also take advantage of any new innovations that may now exist due to the introduction of new technology.

Now try Activities 5 and 6

6. Pr

So far we h ey are very useful
for recordin s and functions
that should ibe the processes.
Behind the hat appear on the
diagrams. T ASE tool will have
the facility t
The DFD's

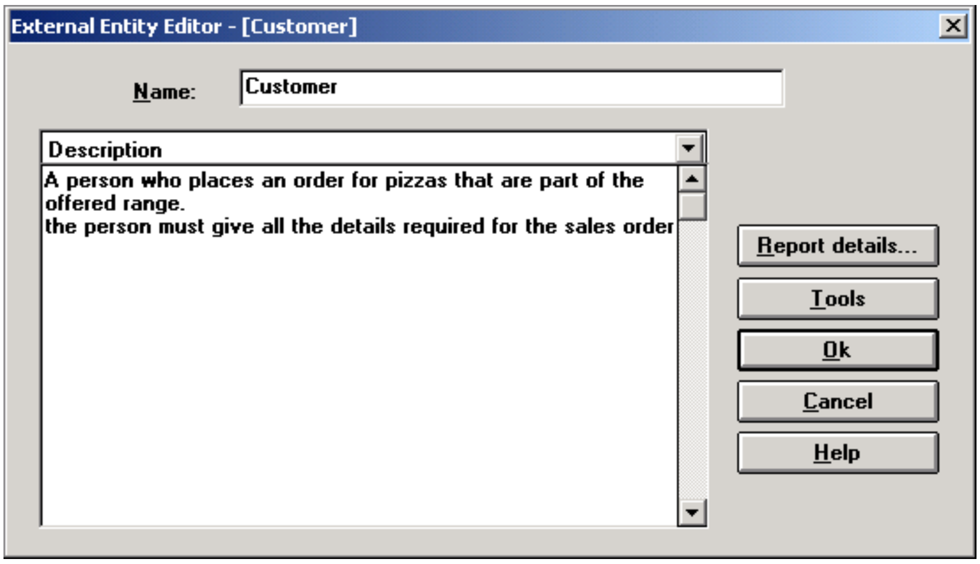

Data Flow Editor - [sales order]

Name: sales order

Description

An order for pizzas.
The order should include:
Date of Order
Sales ID of person taking order
Customer name
Customer address
Customer telephone number
Required delivery date
For each type ordered -
Pizza type (from a list)
Pizza size
Extra toppings required (from a list)
Size of pizza
Quantity required

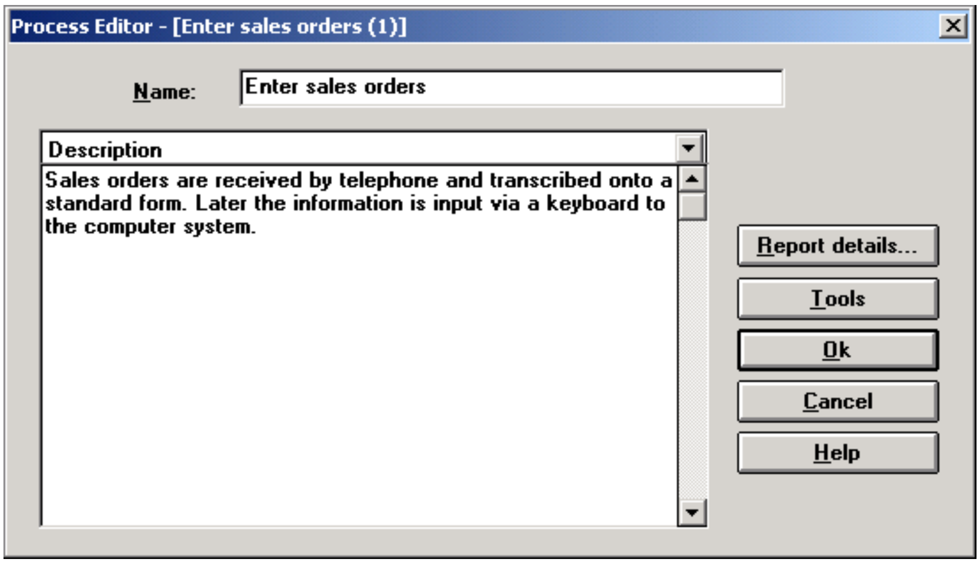
As an example here are some definitions of a data flow, an external entity and a process.


External Entity Editor - [Customer]

Name: Customer

Description

A person who places an order for pizzas that are part of the offered range.
the person must give all the details required for the sales order


Process Editor - [Enter sales orders (1)]

Name: Enter sales orders

Description

Sales orders are received by telephone and transcribed onto a standard form. Later the information is input via a keyboard to the computer system.

7. Summary

Process models are produced to partially describe system that exists and the system that is required. The process models show what is currently being done with the information that is entering the system. The process models do not show how the information is organised but act as a focus for discussion between the developer and the user to determine what the new system should be able to do. The models will also provide a measure during testing to see if the system produced does do what was promised. Typically the processes shown on a data flow diagram will in some way translate into screen designs in the final system. For example a process on a DFD may evolve as a menu item that when clicked opens a window to allow data input, output or simply review.

8. Activities

Exercise 1

The following questions relate to context diagrams:

a) What is it trying to show?

b) How many external entities should there be?

c) How can the context diagram be checked against the document flow diagram?

Exercise 2

a) Represent, using the proper symbols, the following actions:

i) An external entity 'Supplier' sending an invoice to a process 'Deal with payment'. Assume the process takes place in the Accounts division.

ii) A process called 'Update membership' writing customer details to a data store called 'Customers'. Assume the process is performed by the Membership Secretary.

iii) A process called 'Order raw materials' is performed by the Purchasing Clerk. To do this the clerk retrieves details from the 'Supplier' data store and sends the purchase order to the external entity 'Supplier'. The clerk also stores the purchase order details in a data store 'Purchase orders'.

Exercise 3

State which pairs of symbols you are ***not*** *permitted to connect together using a data flow :*

- *External entity – Process*
- *Data store – Data store*
- *External entity – Data store*
- *Process – Process*
- *External entity – External entity*
- *Process - Data store*

Exercise 4

This exercise relates to the case study 'Correspondence Courses' which is outlined below.

Produce (and state any assumptions made):

a) a document flow diagram

b) a context diagram

c) a level 1 data flow diagram

Exercise 5

This exercise relates to the case study 'Animal Park' which is outlined below.

Produce (and state any assumptions made):

a) a document flow diagram

b) a context diagram

c) a level 1 data flow diagram

d) one necessary level 2 data flow diagram

Case–study

Animal Park

The keepers in an animal park look after the feeding of the animals. Each animal is located in a different area of the park. Each area has its own keeper who reports to the head keeper.

The head-keeper maintains a record of the sorts of food that each animal species or type in the park should be fed, and in what quantities. There is no distinction made between different animals of the same species. The keepers access the information so they know what to feed each animal type. Each animal type may be given more than one type of food, and each type of food may be fed to a number of animal types. Each day the keepers will take out the food needed for the animals in their care and record this on the information system. These food types can be perishable or non-perishable according to their shelf life. For example, fresh fruit and vegetables would be perishable where tinned produce or cereals would be considered to be non-perishable.

The office staff keep a track of the food supplies. They monitor which foods are running low every two or three days and draw up a list of that which needs to be ordered. In order to help them, the information system contains details about re-order quantities and re-order levels. Sometimes they may need to readjust their re-order levels.

A number of suppliers are used, and their names, addresses and telephone numbers are kept in the system. Because of the large quantities required and the difficulty of obtaining some foods at certain times of the year, there is more than one possible supplier for each type of food. Most of the ordering is done via the telephone. A standard order form is then created. The order form usually contains details of more than one food type to be ordered from a particular supplier. The order also contains details of the date of the order and what quantities are required for each food type.

When deliveries are received, the keepers check the delivery note against the goods received, amend it if necessary and pass it on to the office. Here it is checked against the orders placed. It they agree, this is recorded in the system. The office staff check the received orders against an invoice sent by the supplier. If they agree, payment is be made. Any discrepancies are taken up with the supplier, and the supplier's response is noted in the system. Most suppliers send an invoice each month.

Exercise 6

This exercise relates to the case study 'Doctors' Surgery' which is outlined below.

Produce (and state any assumptions made):

a) a document flow diagram

b) a context diagram

c) a level 1 data flow diagram

d) one necessary level 2 data flow diagram

Case–study

Doctors’ Surgery

A doctors’ surgery consists of five doctors a receptionist and a manager. They need an information system to help them to run the facility.
A patient may ring the surgery to make an appointment with a doctor. Each patient nominally has a doctor associated with him or her but they may often opt to see any doctor in the surgery that is available. The receptionist sees which doctors are on duty on which days and offers appointment alternatives from which the patient may choose. If an appointment is not available within a short time and the patient must be seen quickly they are asked to attend an emergency surgery that takes place every evening between 5 and 6 p.m. The appointment can be 5, 10 or 20 minutes long, dependent on the reported reason for seeing the doctor. This reason is recorded on the system. Sometimes patients ring to cancel appointments. Appointments may be made for up to six weeks in advance. Appointments that are more than 3 weeks old are automatically deleted from the system. Some appointments are for a doctor to go and visit a patient at home when the patient cannot come to the surgery. Every day one of the doctors is available for home visits in the afternoon.
A record is kept of each patient and the treatments they have received for any ailments they may have had. Here are recorded many details such as allergies, details of which drugs patients have been administered in which quantities and when. Also relevant personal details of each patient are recorded. Typically the doctor who sees a patient will want access to this information before deciding on the relevant treatment to give. When the doctor prescribes treatment, details will be recorded in the patient’s record.
Repeat prescriptions are automatically produced by the system and are available for collection at the surgery by the patient. At any time a doctor may suspend or cancel the prescriptions.
Patients may register with the surgery providing the number registered to each doctor is not above a certain maximum. Sometimes patients die or leave the area. In this case the patient is removed from the system and their details are archived. The manager is responsible for dealing with this aspect.

9. Bibliography

Goodland M., Slater C., *SSADM – A Practical Approach*, McGraw Hill, 1995

Kendall K., Kendall J., *Systems Analysis and Design*, Prentice Hall, 1998