Process modelling:
Data Flow Diagrams (DFDs)

1 LEARNING OBJECTIVES

Students should by completion:

- Be able to describe the four elements of a data flow diagram and how they may be shown graphically.
- Understand what 'partitioning' and 'levelling' mean, and how these allow us to represent complex situations in a simple form.
- Know what is meant by a 'context diagram'
- Understand what 'balancing' means and how this ensures that there is consistency between the representation of a situation at different levels of detail.
- Appreciate how using four sets of data flow diagrams allows the analyst to move from understanding the present situation to designing new procedures and information systems
- Understand the difference between logical and physical data flow diagrams
- Be able to create data flow diagrams to represent a described situation using the correct conventions.

2 RELATED READING

The topic is widely covered and an introduction to data flow diagrams will be found in most books concerning systems development. Good examples of this are:

Greasley, A. (Ed.)) (2006) Business Information Systems (3rd Ed.), Harlow, Pearson Education pp 438-445

Curtis, G. & Cobham D. (2002) Business Information Systems: analysis, design and practice.(4th Edition) Pearson Education, Harlow.

Kendall K.E. & Kendall J.E. (2002) Systems Analysisi and Design (5th Ed.) Prentice-Hall International New Jersey.

For a more detailed coverage I believe that the original texts that first popularised the ideas remain the clearest exposition why to use DFDs and how to use the technique. These are:

De Marco, T. (1978) Structured Analysis and System Specification. Yourdon. Chapters 4-7
Gane, C. & Sarson, T. (1977) Structured Systems Analysis: tools and techniques. IST. Chapter 3

Since DFDs became one of the most widely used process modelling tools the topic has been included as part of many methodologies for systems development, including SSADM. Any text describing the methodology will include some description of DFDs, including:

Downs, E., Clare, P. & Coe, I. (1992) Structured Systems Analysis and Design Method: application and context. Prentice Hall, Hemel Hempstead.
Weaver, P.L. (1993) Practical SSADM: A complete tutorial guide. Pitman, London.

3 THE NEED TO DOCUMENT PROCESS ANALYSIS

If we are trying to understand an organisation (or part of an organisation) with a view to improving its operations and/or providing computer support then one very important part of our investigations will be to find out about the business *processes* that happen within the organisation. This is done by discussions and interviews with the people who work in the organisation, by observing working practices, and by following the history of typical transactions.

Once one has gained knowledge about what is done, how it is done then one needs to document this. We do this

- So that we remember all the detail of what we learn
- As a way of coming to a shared agreement about the situation
- As a way of detecting omissions or misunderstandings in our knowledge
- For the benefit of others who may need to take over the work or continue the work into later stages of development
- To provide the basis for future discussions about how things may be re-engineered.

4 DATA FLOW DIAGRAMS

Data flow diagrams (DFDs) are widely used simply because they are a good means of doing this. They are easy to learn, readily understandable and can therefore be shared with people in the problem situation.

A data flow diagram shows the business processes that occur within an organisation and the way in which data travels between these processes. Both manual and automated processes are shown, so data flow diagramming leads us to look at all the ways in which data is used within the system, and looks at the situation from the point of view of the business. A data flow diagram therefore is a model of the system as seen from the point of view of the processing and communication of data. The DFD is a working document used both as a means of communication with the users and as a constantly changing record of the analysts' current knowledge.

There is a disadvantage associated with the emphasis on data in that where the business is primarily concerned with the flow of goods and products, e.g. in manufacturing, the DFDs will not capture all the richness of the processes involved, other than where data is somehow generated or used.. So where there is a physical transportation of goods this will only be indicated by the accompanying transfer of delivery notes etc or by the resulting communications, say between the delivery driver and the warehouse manager.

There are no universally agreed diagramming conventions for data flow diagrams and that you will find that different authors will employ different diagramming notations in their discussion of DFDs. The fundamental ideas are though the same in all of the readings.

The diagramming used here are those most commonly used and are consistent with the SSADM methodology.

5 THE ELEMENTS OF DATA FLOW DIAGRAMS

Data flow diagrams are built up from only four different elements, namely:

- Data Flows
- Sources and Sinks
- Processes
- Data stores

5.1 Data flows

Can be thought of as being 'pipelines' through which 'packets' of data flow. They are shown as a labelled directed vector as shown in Figure 1.

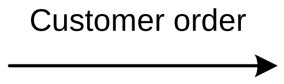

Figure 1.

5.2 Sources and sinks

Every time we do analysis we are faced with a trade-off between trying to understand everything and the limitations of time and effort so we have to make some choice as to exactly what we are going to investigate.

In systems terms we must draw a *boundary* to separate those things, which we are going to study in detail and those that we are not. In many cases this boundary is drawn for the analyst, perhaps because we are working for one department and cannot intrude upon another or because our client has strong ideas about what we should be looking at; in other cases we draw the boundary ourselves by deciding which areas are most relevant to our client's perceived problems. We will though need to take account things outside the problem situation, in terms of the data that flows across the boundary.

Both sources and sinks are entities, which lie outside the system boundary, which produce or receive information for the system that we are studying. As we cannot or are not going to change their operation we may regard them as 'black boxes'

Sinks and sources are represented as shown in Figure 2 and may be regarded as the original source or final destination of some piece of data.

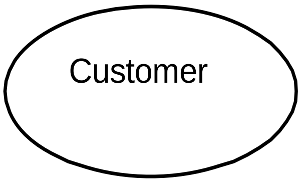

Figure 2 A sink/source

Note that because these lie outside the system boundary they appear as actual 'real world' entities such as customers, the government etc. Had the system boundary encompassed these entities (i.e. they were part of our system) then they would have *only appeared in terms of the processes which they perform*. Because they lie outside the system boundary they are only of interest in as much as they interact with the system; anything else which they do, and any interactions between them, are not of interest to us and not shown on our data flow diagrams.

5.3 Processes

The basic business processes of the organisation are represented as transformation processes which transform input data flows into output data flows and are shown on data flow diagrams as rectangles into which and from which data flows travel, as in Figure 3.

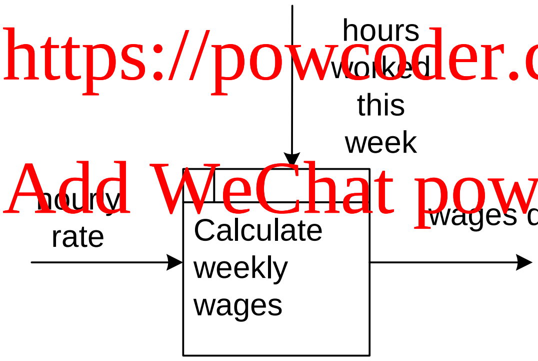

Figure 3. A process

Remember that this is a *data flow* diagram, and focuses upon the movement and processing of data only. A business process may, of course, involve the movement of physical goods such as products but these are not directly shown on the DFD. If the shipped goods were accompanied by a delivery note though, that would be shown.

5.4 **Data stores**

Data stores are any place where data temporarily resides and comes to rest. Some common examples are magnetic disc files, filing cabinets, your address book in which you keep phone numbers or the ring binder in which you keep your course notes.
Even if the data is only temporarily halted, for example a clerk collects together the day's completed forms on the edge of his desk until he takes them all over to someone else for the next stage of processing at the end of the day, then a data store is being used.
Data stores are depicted as shown in Figure 4.

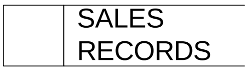

Figure 4.

When we show a data store being updated we only show the net flow of data, i.e. even though physically we have to read a file before updating it we only show the one arrow going into the data store. If however we read the file as part of the business process and then later update it we show the dataflow as a two-way flow.

It is intersting to note that the original promoters of DFDs as an analysis tool argued that it was not necessary to labels data flows going out of or into a data store since they could be aassumed to have the same contents as the data store itself. This is now not held to be true, since we may retrieve from the datastore only a sub-set of the data items therein, and because we wish to define precisely the data items in every dataflow in order that we can combine the results of DFD diagramming and other techniques such as entity-relationship modelling.
Figure 5 shows how the individual symbols may be used together to define a particular piece of business processing.

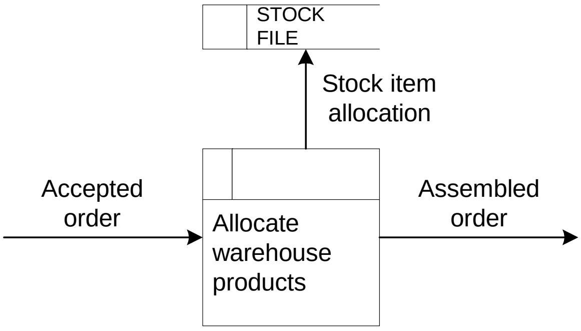

Figure 5.

6 PARTITIONING AND LEVELLING

If we drew a single data flow diagram showing every business process that went on in a company, in detail, then we would end up with an enormous and incomprehensible picture. A system of any size will be far too complex to comprehend in one go.
So DFD diagramming uses the principle of 'divide and conquer', breaking down each complex process into its constituent sub-processes and gradually looking at the system in greater and greater detail.
These sets of levelled DFDs provide a **TOP DOWN** picture of the system, where each level expands the detail given about the level above.

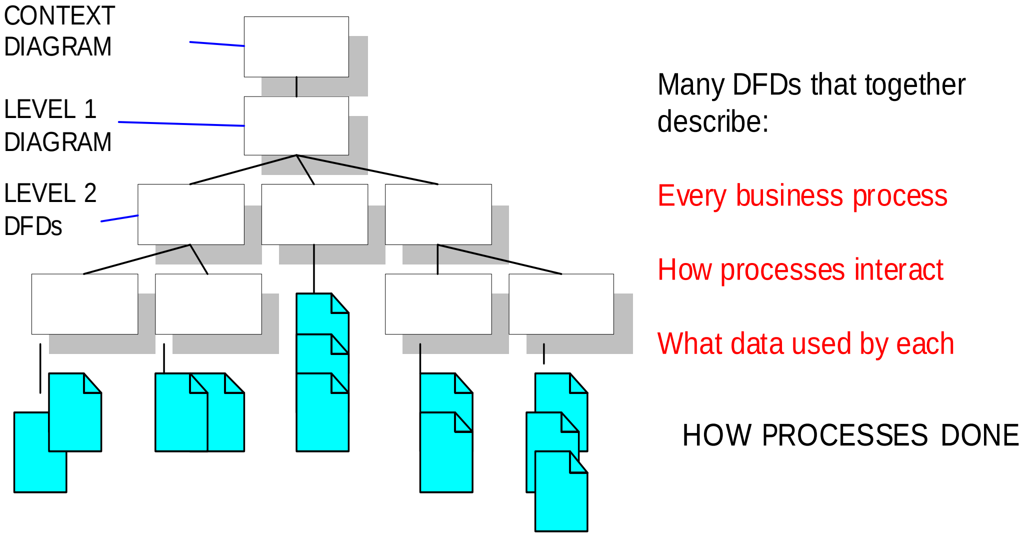

Figure 6.

We start at the very top level by drawing our **CONTEXT DIAGRAM**. This shows a single process (the system under review) and a number of inputs and outputs.

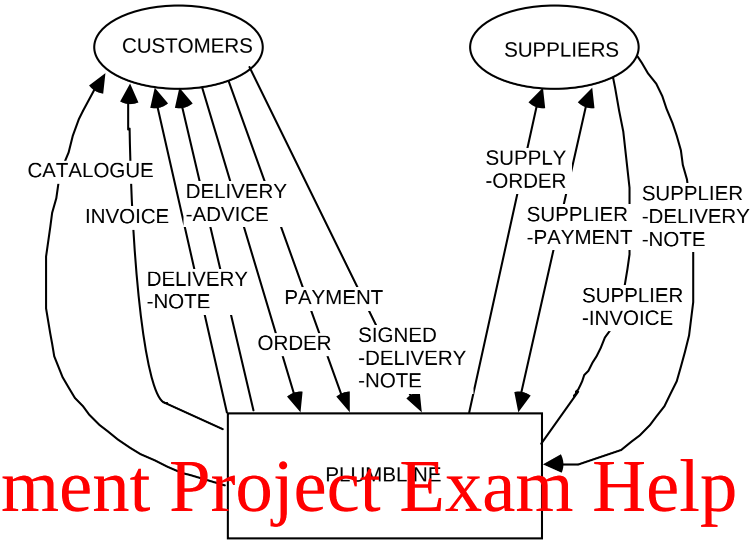

Figure 7.

The next most detailed level is the Level 1 diagram, which shows the main processes going on within the system.

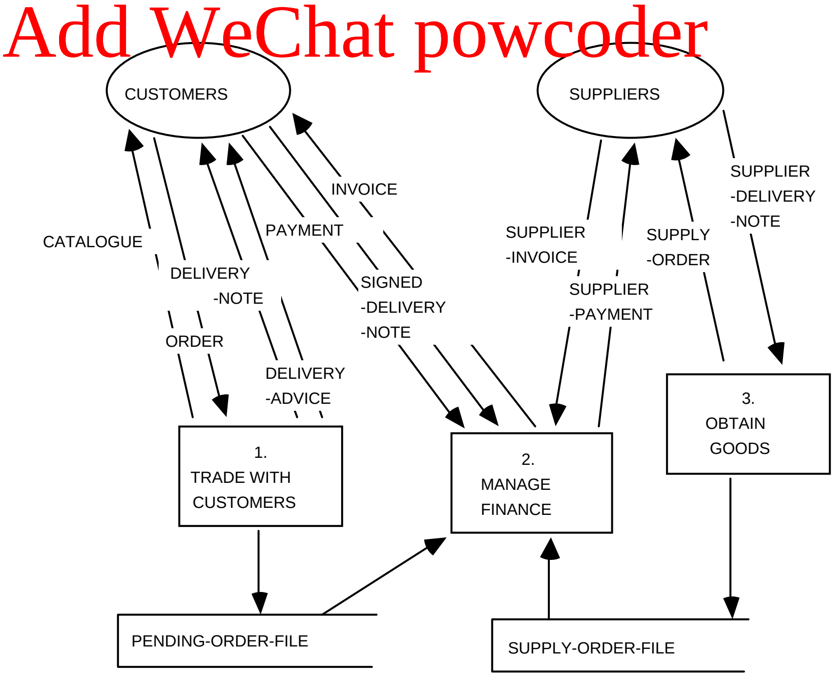

Figure 8.

The Level 2 diagrams will then show the sub-processes going on within each of these main processes, and so on down through succeeding levels. Figure 9 below for example shows a more detailed view of the sub-processes going on within "Manage Finances". We rarely need to go down more than 6 levels.

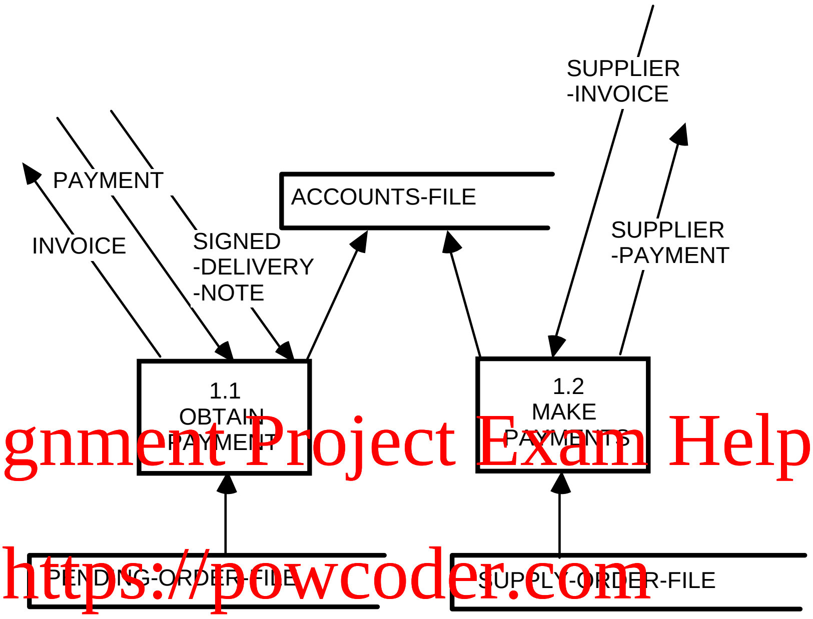

Figure 9.

When we break a process down into sub-processes we talk of the higher level process as being the FATHER of the CHILDREN sub- processes. In order to be able to keep track of the relationship between the various processes and sub-processes it is usual to adopt a decimal labelling system. Thus when we decompose Process 1 we label its children as 1.1, 1.2, 1.3 etc., as in the examples.

We have to decide when to stop this process of decomposition, and the rough rule for this is that we stop when we can describe a process on one side of a sheet of A4 paper. Those processes, which cannot be usefully decomposed further, are called functional primitives. There will eventually be one "mini-specification" for each functional primitive, which will describe the process in detail. Later still these mini-specs might form the basis of Program Specifications, the instructions to programmers about exactly what each program should do.

7 BALANCING

When a process with two inputs and two outputs is broken down into sub-processes the data flow diagram showing the sub-processes must have exactly two inputs and two outputs. If it does not then you have something wrong so go back and check. The situation below would be cause for concern. Consider Figures 10 and 11 below.

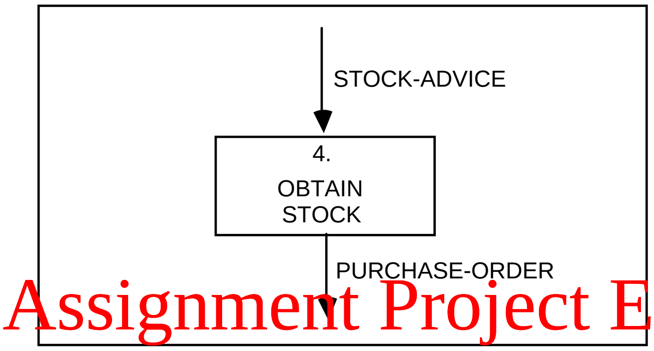

Figure 10. Part of a Level 1 diagram with balancing error

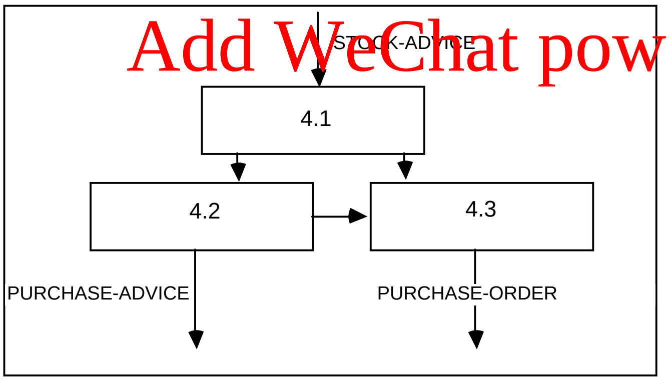

Figure 11. Level 2 diagram

The error is that the diagrams do not balance, i.e. they tell a different story about what information flows in and out of the process 'Obtain stock'. In order for the diagrams to balance the Level 1 diagram should show an additional data flow, "Purchase-Advice"

8 USING DATA FLOW DIAGRAMS IN ANALYSIS AND DESIGN

In practice we do not draw a single data flow diagram but **four** different data flow diagrams, as in Figure 12.

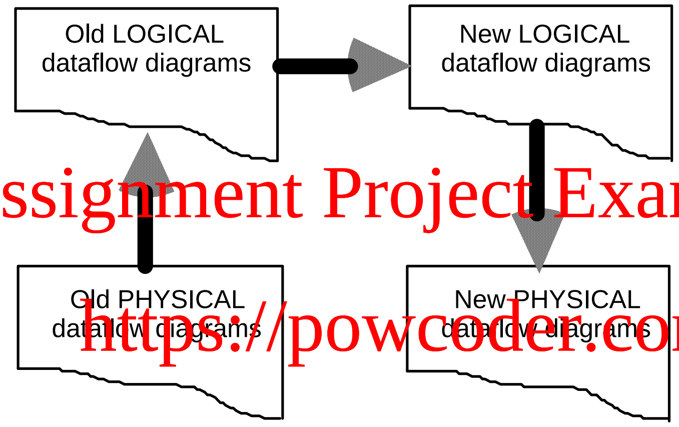
Old LOGICAL dataflow diagrams
New LOGICAL dataflow diagrams
Old PHYSICAL dataflow diagrams
New PHYSICAL dataflow diagrams

Figure 12. Four types of DFD used

Firstly we investigate the present business situation and draw a data flow diagram which describes this, this being known as the **present physical system** diagram. Since this diagram is a description of how the business operates at the moment it will be rich in physical details, for example naming the particular forms which are used to pass information around the system or the personnel who currently perform certain procedures.

This diagram will be built up from extensive interviewing and observation of the business situation. The diagram itself can be easily used as a basis for discussion with the users and they can be taken through it step by step in order to check that it is a correct record of what happens at present. Figure 13 shows part of the physical data flow diagram drawn for a builder's merchants company. Note that it shows the name ("PLUMB-07") of the actual paper form currently used.

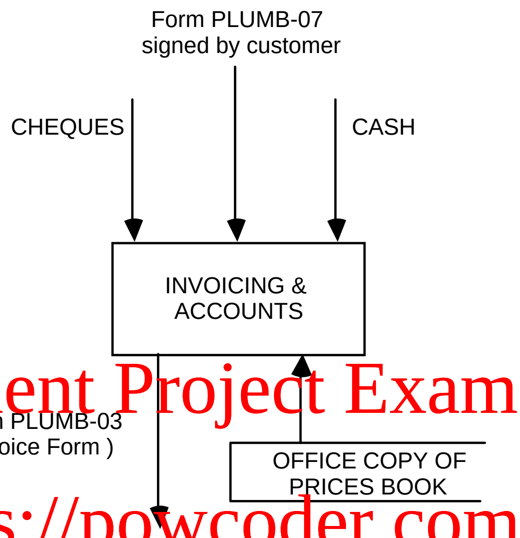

Figure 13.

Once we have created a picture of the present physical system then the next stage is to 'logicalise' this data flow diagram, i.e. to remove all the physical details. We need to do this because in order to really understand the business we need a picture of what is going on freed from all the irrelevant details of who is presently employed, which particular forms are used etc.; we want to understand the flows of information around the system but whether that information is presently passed by means of a green coloured form called a P230 or by a telephone message is really somewhat irrelevant.

The diagram which we obtain from this process is a data flow diagram of the present logical system, and the previously drawn diagram might now appear as in Figure 14. Here we show not the name of the PLUMB-07 form or that it has been returned signed, but the data conveyed by that signed form, namely that the customer has confirmed delivery.

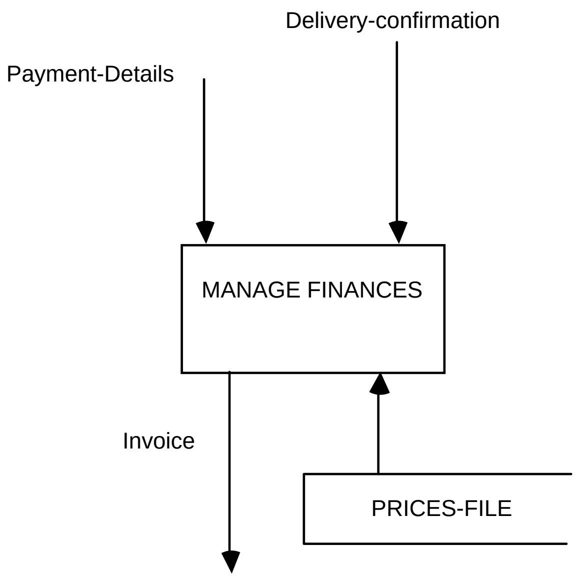

Figure 14.

Moving from the physical to logical diagram is not however merely a question of changing the names of processes and data flows, the structure of the diagram may itself change. The diagram of the present logical system may be harder for the users to understand than that of the present physical system since it does not show details of the people, forms etc. with which they are familiar. However it can still be used in discussions if the analyst is careful to interpret it for the users, perhaps talking them through it and in doing so adding back some of the physical details to which they can relate.

The diagram of the present logical system is a record of the analyst's understanding of the present situation and as such is the basis for his bringing about improvements. The next stage is to decide how to improve the business operations, or to be more precise, to decide what the new system should do. Again we are working at the logical level and do not at this stage consider how particular processes or transfers of data will be achieved or even which parts of the system will be computerised. We are considering the new business system, not the new information system. At this stage the analyst often finds that he is not free to consider changes to all of the old system because of financial or organisational limitations. That part of the old system that is open to change is defined as 'the Domain of Change'.

It is very important to realise these decisions about what the new information system will (or will not do) are not the responsibility of the analyst but of the users and that the role of the analyst at this stage is to help and advise rather than to decide what is best for others.

Once the analyst has a model of 'what' the new system will do he can then turn to designing 'how' it will do it. In defining the new physical system he enters the most creative stage of the analysis and will consider which parts of the system should be computerised (known as defining the 'man-machine boundary'), what type and size of machines will be required, what forms will be needed, who will perform particular processes, how the users will be organised administratively and numerous other design questions. This obviously requires not only considerable technical knowledge and experience but also close consultation with the users and user management.

9 SOME GUIDELINES FOR DRAWING DFDS

Drawing good (that is to say useful) data flow diagrams takes practice but their are some practical guidelines which we can follow and some common mistakes which we should avoid. The important thing to remember is that time spent in getting the diagrams right is never wasted and will save vastly more time in the long run.

- Give everything meaningful and descriptive names. Not only will this help keep things clear to you but it will be invaluable when you show the diagram to the users.
- As a general guide try to name your processes as two words, a verb followed by a noun e.g. CALCULATE WAGES or RECORD PAYMENTS.
- If you cannot think of a good name for a data flow then WATCH OUT! It usually means that you've got a couple of separate data flows combined or at least are not sure precisely what data is flowing here.
- Similarly avoid naming any data flow with vague names such as WAGES - INFORMATION, WAGES - DATA, INPUTS or anything else which shows that you do not really know what is going on.

- Make sure that your diagram really does show things from the point of view of the data. It should not show control information and must NEVER show decisions.
- Try to make sure that all the processes shown on a single diagram are at roughly the same level, i.e. do not have a process called CALCULATE AGENTS' QUARTERLY COMMISSION on the same diagram as one called MAINTAIN COMPANY ACCOUNTS. If you do then you risk either confusing the user or getting bogged down in the detail too soon.
- Check your data flow diagrams for information traps. These are files to which one or more processes write but no process ever reads. Is this really what happens? If so then why bother saving this information if no one ever uses it.
- Look at each data store shown. Is it being updated anywhere? Is all the information being updated? If not should it be?
- Make sure that your diagram does not show too much physical detail unless you want it to in order to talk more easily with the user.
- There should be no physical detail shown when working at the logical level. The data flow diagram at this level will not distinguish between manual and automated processes.
- Make sure that your diagram looks reasonable and that the outputs from any process can be created from the inputs shown.
- Do not bother showing error cases until the very last versions of your diagrams since they only confuse the issue.
- Remember that your diagram is always going to change. Encourage this by talking through it with the user and other team members as much as possible. These diagrams are your equivalent of the architect's model so make sure that you are building the house that your client wants and that it will not fall down.

10 COMPARISON WITH OTHER DIAGRAMMING CONVENTIONS

As data flow diagrams are only one of a range of diagramming conventions which one may encounter it is important to differentiate between the different conventions:

- It is different from the **entity-relationship (E-R) models** used in Data Analysis in that these show the structure of the data but nothing about how it will be used.
- It is different to a **flowchart** in that a data flow diagram does not show control or decision making.
- It does not show time. This is sometimes a difficult point to appreciate but do not imagine a data flow diagram as showing the route which any particular piece of data through the system; the diagram is a picture of a system in operation, not a snapshot of the system at any single moment in time. It is intended to show the relationships between the processes and the information used in a system, and no more than this should be read into data flow diagrams. This is in fact one of the weaknesses of DFDs compared to other techniques as **Entity Life History Diagrams**.
- It does not give any information of behaviour, such as given by **object-oriented methods.**
- DFDs are focussed on the essential things that must in some way be done (what they call the logical activities) rather than the detail of the sequencing or detail such as who will do them. Conventions such as **swimlane diagrams** instead focus on the specifics of a particular procedural design, showing the particular chosen route for information to pass between organizational roles.