STRUCTURED ANALYSIS

D.T. Ross

Rapporteurs: Mr. R. Gimson
Mr. L. Marshall
Dr. S. Shrivastava

Part 1

The Structured Analysis Maxim

The following is called the Structured Analysis Maxim:

> Everything worth saying about
> Anything worth saying something about
> Must be expressed
> In six or fewer pieces

If you absorb and believe what the Maxim says, you will already believe everything I am going to tell you. If you find it natural to achieve that understanding, then you are one of the lucky people who have already been practising Structured Analysis for many years.

To apply the Maxim I must first of all have something which is worth saying something about - on any subject matter. Then the key word is 'everything'; everthing about that subject matter must be expressed in six or fewer pieces. What the pieces consist of depends, of course, on the subject matter. The point is that you use preferably not one piece (since not much progress has then been made in the analysis), not usually two pieces, but generally three, four or five, and never more than six pieces.

Provided you have a worthy subject matter, merely chopping it into three or four pieces has not normally exhausted that worth. So every one of those pieces in turn becomes a worthy subject matter worth saying something more about, and the Maxim applies again. Everything that makes up one of the pieces is broken again into its own pieces, until pieces are reached that have no further worth and which are not broken down further.

Why is it necessary to have this chaining of 'everything' as I do this recursion? Well, if it were not so, on breaking an item into pieces something would get lost and I would only have a part of the original. Only by requiring that at every stage I really have everything, and that the parts continue to tie together to make the same whole, does the analysis work. The difficulty is to find a notation - a set of mental semantics with a means of writing analyses down and communicating them among people - that will live up to the Maxim and make it come true.

There are no limits to the subject matter that can be structured in this way. The reason that the structuring works and is beneficial is that you are making ideas into bite-sized chunks. You can understand anything better if you understand how it is composed out of the meaningful pieces which go together to make it up.

Structured Analysis and Design

From now on I shall talk about the Structured Analysis and Design Technique SADT^{TM} in the context of system design and building. (SADT^{TM} is a registered trademark of Software Technology, Inc., Waltham, Mass., U.S.A.) SADT is an integrated approach to performing systems analysis and design extending from requirements definition, which is the stage before system specification, through to the installation and maintenance of the system. It produces documentation concurrent with every stage of development. Most important it allows communication between not only designers, analysts, and users, but also the managers and others peripherally affected by the project. It assures quality and configuration control both by keeping track of the current status of all parts either of the system or of the documentation, and also through the discipline of continuous review and approval. This is similar to the concept of egoless programming advocated by Weinberg, but in our case we would like to have egoless analysis and even egoless management. The key fact is that everything is visible and understandable and cast in such terms that those people who are concerned feel that the appropriate way to work is openly, looking for improvement to their contribution to a team effort.

Structured Analysis also seems to scale rather well, so that without large changes in the method it can be applied to large projects as well as small ones. It is applicable in the early stages of requirements definition, analysis and design, and with further refinement (because some of the notation becomes less appropriate), also to the remaining parts of the system life cycle, so that these parts can be represented in a compatible structured form. However, it is primarily its use in the earlier stages that we are talking about. It works for both software and hardware, and also for both human-only and mixed human-machine systems.

The SADT Model

We make a model of the subject matter (that is, the 'system') consisting of a top-down hierarchic set of 'blueprint' drawings in which the high level over-view presents the whole subject, the 'everything', and each diagram shows only a limited amount of detail in easy-to-grasp units. It will turn out to be important to decide in which direction this decomposition takes place. It is important to have a well-specified viewpoint of the subject matter. In some context, thinking about some subject, you pick a viewpoint from which you are going to look at the system. Within that viewpoint you have a certain purpose in mind for exposing its structure to make it more understandable. This means that with respect to the purpose some questions are going to be most important, and then within the context of those questions another layer of questions can be asked. The hierarchic layering is achieved by putting the most important questions first and using Occam's Razor to separate them at each stage, giving a worthwhile modularity. As in the maxim, in the graphical form each of the pieces is worthy in itself and is further broken down (see Figure 1).

Blueprints

The purpose of the 'blueprint' notation is to allow the expression of different viewpoints for different purposes. The term 'blueprint' is used because they serve the same role as blueprints do in manufacturing, and have the same nature of completeness. When building a house there are different sets of blueprints for, say, the physical structure of the building and for the plumbing, allowing different viewpoints for different parts of the whole, yet the notation allows everything to easily fit together.

The key graphical forms of Structured Analysis diagrams are boxes, which for example show activities, and arrows, which show the interfaces of a box and clarify its action. Boxes always have four sides, each side having a fixed meaning. The left side always means input, and the right side output, and within the box is written the name of the concept being described (see Figure 2). In that sense they look similar to HIPO charts. However, the major difference is that the top side of a box means control and the bottom side is not an interface between boxes, as in HIPO, but is the mechanism by which the box is implemented.

Considering the decomposition of the system into parts, each of which is a process or activity, if the action described by a box corresponds to a verb, then the arrows correspond to nouns. Alternatively, if the box corresponds to a noun, the arrows correspond to the verbs. Taking an individual processing box,'input is converted into output under the influence of control, by the mechanism'. The control data constrains how the input is modified to produce output, or how it can be used without modification as the output.

The arrows can be thought of as conduits with cables in them, or rather hierarchically nested cables. Where on a diagram you may see a single arrow going between two boxes, inside that arrow is further decomposition. So, in Figure 3, vegetables consist of root crops and leaf crops, each of which may be handled differently by the growing and selling functions. As the boxes are decomposed, so also are the arrows decomposed; the arrows are just as abstract as the boxes.

Modularity

Now let us see how we can get some order out of the idea of successive layers of abstraction. What do we mean by a 'modular system'? A modular system is one which is decomposed into pieces in order to achieve some purpose. Synthesis is composition and analysis is decomposition, and Structured Analysis is structured decomposition. A decomposition is required from which you are always able to recompose the object. The human mind can understand any amount of complexity (nobody has yet found a mind that was really saturated!) provided it is presented in small easy-to-grasp pieces that are structured together to make the whole. The relationships among the modules are shown explicitly and at any level of detail you can see both the pieces and how they interrelate to make the whole. So in the model any particular aspect that you want to understand can be and must be understood in terms of its entire position within the whole system. At each point there is a bounded context in which you can 'zero in' as you go down the hierarchy to the required point (see Figure 4).

Now let us see how the boxes and arrows go together (Figure 5). The arrows go from box to box, but denote constraints between boxes rather than flow. This can be seen clearly from the next important aspect of the Structured Analysis approach. Everything has to be considered from two different viewpoints, 'things' and 'happenings'. These are the activity (that is, happenings) and data (that is, all physical or abstract data objects) viewpoints. In terms of information processing systems it is the system processing and the system data which makes these two views (Figure 6). The same box notation is used for both viewpoints.

In the case of an activity box (Figure 7), the input and output data are easily seen as the objects being processed. For example on a factory production line the inputs are the bolts and screws and the output is the assembled object. This is carried out under the control of the drawings or manufacturing instructions, and the mechanism involved in carrying it out is the man or machine doing the assembly. In the case of a data box (Figure 8), the arrows represent the activities which either create that thing or use that thing, or which control its creation or use. As an illustration, in Figure 9 we can see that the data box in one view becomes the data arrow in the other view, and similarly the activity arrow becomes the activity box.

The data notation is the dual of the activity notation, but however, the data model is not the dual of the activity model. It is instead a fresh but complementary start. It provides a separate look at the same reality. In activity diagrams, data is clustered by interaction of activities, while in data diagrams, data is clustered by the purpose it serves.

By having these two views it is possible to take a different look at the system. The system is modelled afresh in both ways, so that the system description is complete in each, only differing in the focal point of the boxes being either activities or data. Once two views have been obtained they can be cross-referenced and integrated, enabling each to be checked for completeness against the other. A complete model is one which has the two views related together, giving a complete description of the system.

Why do I require these two views? Why can't I continue the decomposition using a single view? Well, if a complete decomposition is attempted, you tend to lose focus of the description before a complete decomposition has been made. The reader who is following the diagrams also runs out of his ability to follow and understand what you are talking about. Huge hierarchic trees in a decomposition are no longer understandable, because the purpose that they serve becomes less related to the original high-priority questions in the top-level diagrams. After progressing down through a few levels, the original purpose of the description has gone out of focus. What you should do instead is take a new view and start decomposing again. What were previously lower priority considerations in the first viewpoint may turn out to have a high priority from another viewpoint, and so again can be sharply in focus at the higher levels of description. By having separate viewpoints, nothing gets so far down in the hierarchy that you get really confused trying to get to it. If it does, the analyst has not done a good job at picking these multiple viewpoints and purposes.

Cross-Referencing

Cross-referencing the data and activity views enables errors and oversights to be discovered and also uncovers places where the decomposition hasn't been taken far enough or where a different factoring is required.

The classification process, in which the activity and data models are cross-referenced, is as follows. For any individual arrow segment in one model it is known from its position and identifier what notion it is intended to represent. Moving to the other viewpoint, the arrow will appear somewhere as a box, since it is part of the model and the model describes everything. The notion must fit into the first, most general box because everything does. Which of the three, four, five or six components of that box does it fit into? Say it fits the second. That box also is decomposed and the notion is found to fit one of its components. By following down through the decomposition tree, the notion is classified as fitting into a particular component at each stage. On getting down far enough, it is found that a stage comes where the notion will not exactly fit one of the components, but is related to two or three components; it doesn't exclusively fit any single one. So when it won't go any further, even though the bottom of the decomposition has not yet been reached, the classification has to stop. By deriving a unique number corresponding to the route taken down the decomposition tree, the box at which the classification stops can be named, and that number is written back onto the arrow denoting the notion in the first model. A similar technique is used to relate the arrows of the second model to the boxes of the first. Note that it is not a process of mapping arrows onto boxes, but rather one of classifying the subject that the arrow represents into boxes. The decomposition levels of the arrow in one model and its corresponding box in the other will never correspond exactly.

Viewpoints

A viewpoint is selected for the problem as a perspective from which the whole problem will be studied. The choice of a viewpoint means that certain aspects will be emphasised and others obscured. The problem may be studied from more than one viewpoint, but separately for each one. Figure 10 shows a subject being decomposed according to some viewpoint, and demonstrates the numbering of the components in the decomposition. Down at the lower levels of decomposition things start to go out of focus with respect to that viewpoint. When doing classifications, they will end up in the meaningful parts only if the two models are pretty much in balance. If everything gets stuck up in the top part of the tree and nothing is able to classify at lower levels, it indicates that you did too much modelling in the data domain. For example you decomposed the data right down into the bits from which it is composed, but in themselves the bits are not meaningful to the model.

The objective is to cover the subject with several viewpoints so that it is studied completely by combining analysis to an appropriate depth from each (Figure 11). Each viewpoint covers the whole topic but no viewpoint goes so deep that it encounters detail which is not interesting. When it comes to choosing viewpoints for systems, we can consider it from several person's points of view, each having their own particular idea of the parts which they see as most important. For example a system can be considered from the viewpoint of the user, builder, operator, maintainer, manager, and seller. The same system contains all the same features, but the importance will be different. All the viewpoints can be tied together, though, to give a complete picture of the system. The key is to have a technique for getting onto paper the appropriate worthy information, this being the objective of Structured Analysis.

SADT MODEL

- AN ORGANIZED SEQUENCE OF "BLUEPRINT" DRAWINGS
- A HIGH-LEVEL OVERVIEW PRESENTS THE WHOLE SUBJECT
- EACH DIAGRAM SHOWS A LIMITED AMOUNT OF DETAIL IN EASY-TO-GRASP UNITS
  - A WELL BOUNDED SUBJECT
  - LIMITED AMOUNT OF DETAIL

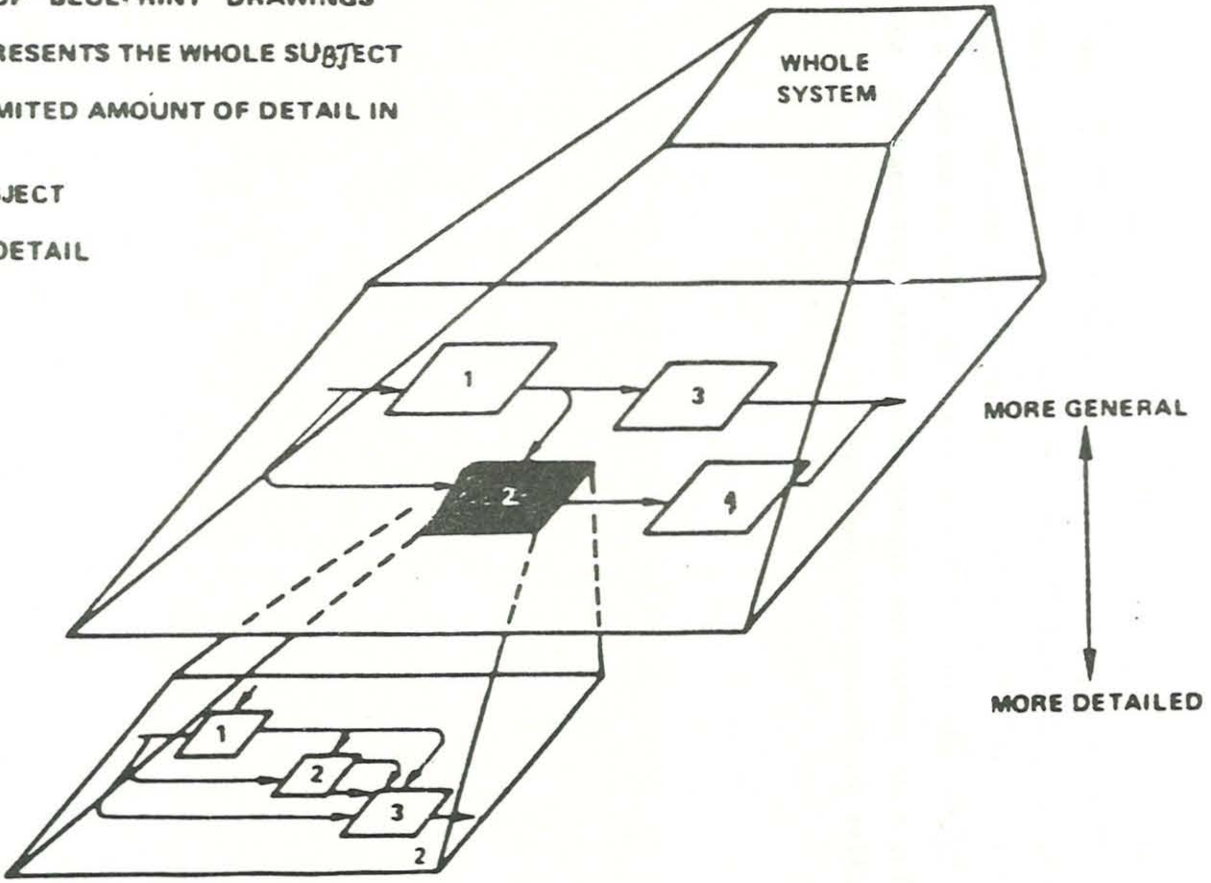


Figure 1

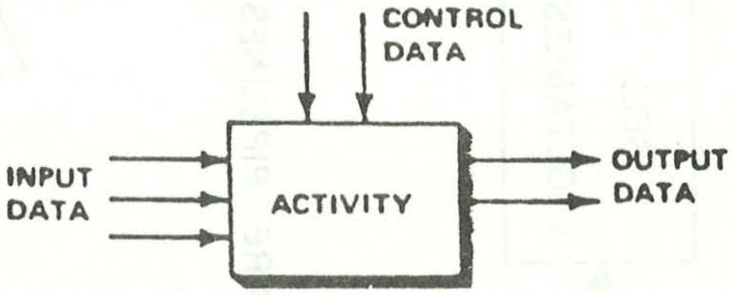
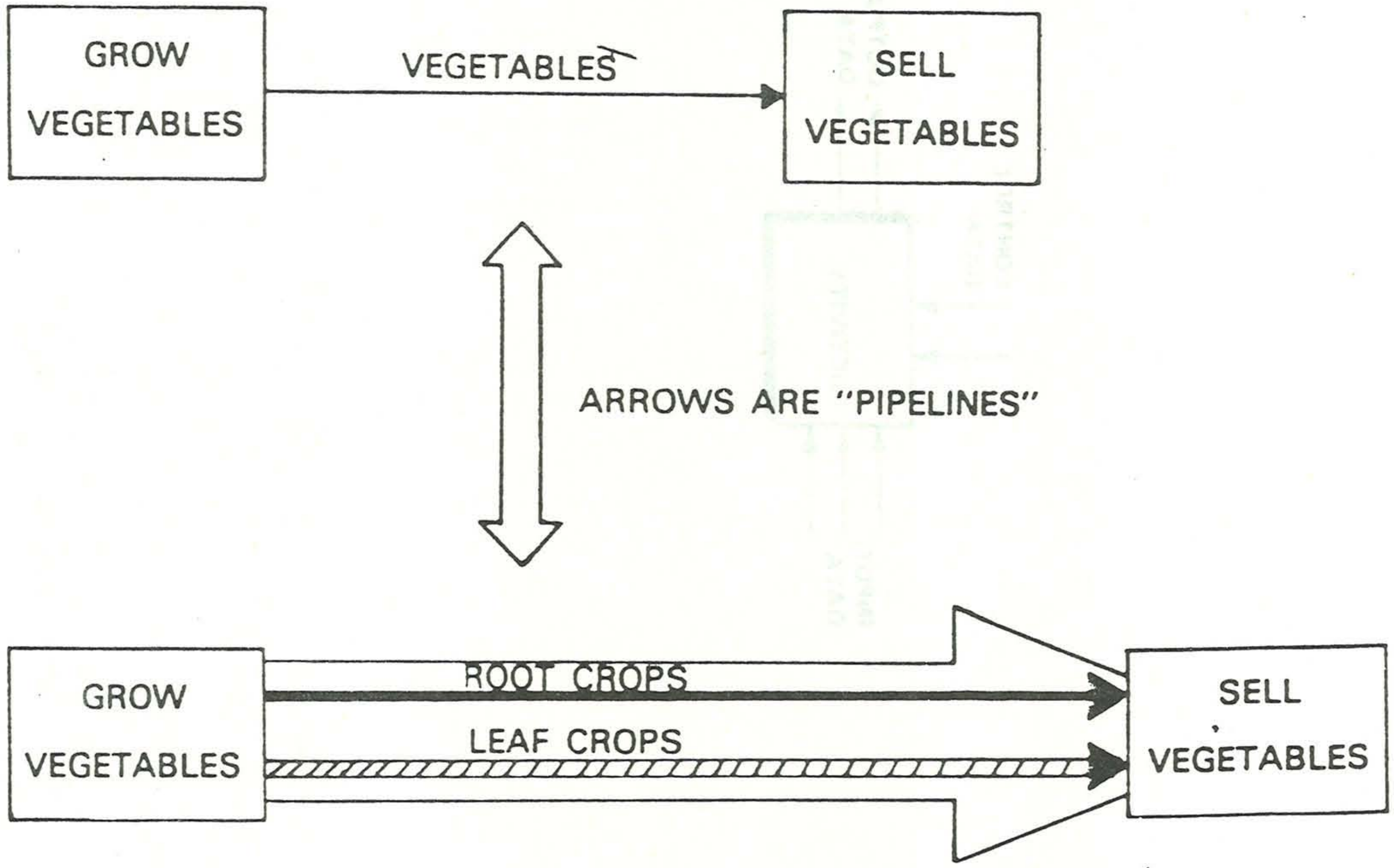


Figure 2 SADT Activity Box.

Figure 3

ARROWS REPRESENT CLASSES OF DATA



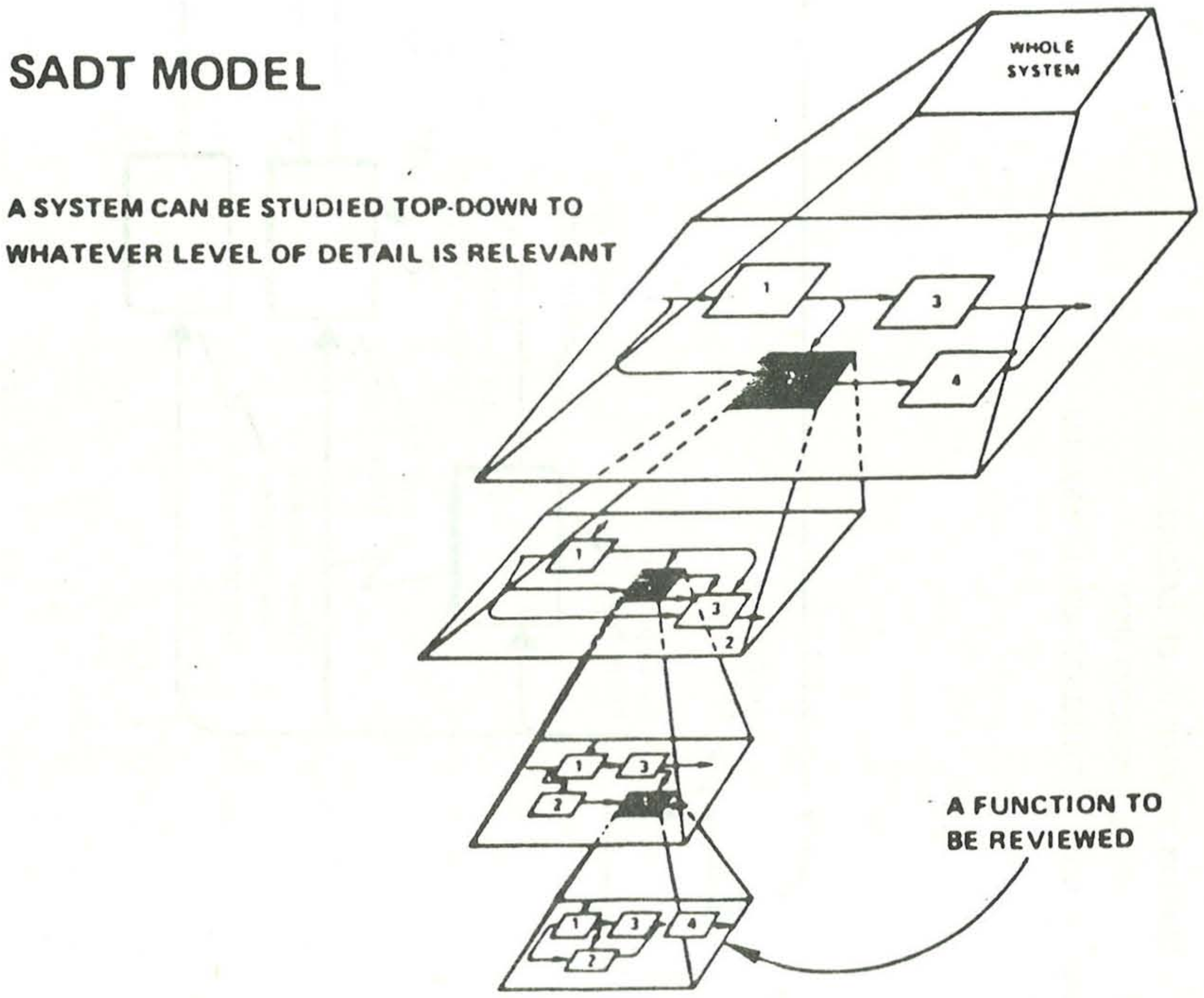

SADT MODEL
A SYSTEM CAN BE STUDIED TOP-DOWN TO WHATEVER LEVEL OF DETAIL IS RELEVANT
WHOLE SYSTEM
1
3
4
1
3
2
1
3
2
1
3
4
2
A FUNCTION TO BE REVIEWED

Figure 4

A DIAGRAM IS MADE UP OF LABELLED
BOXES CONNECTED WITH
ARROWS TO SHOW CONSTRAINTS AND INTERFACES

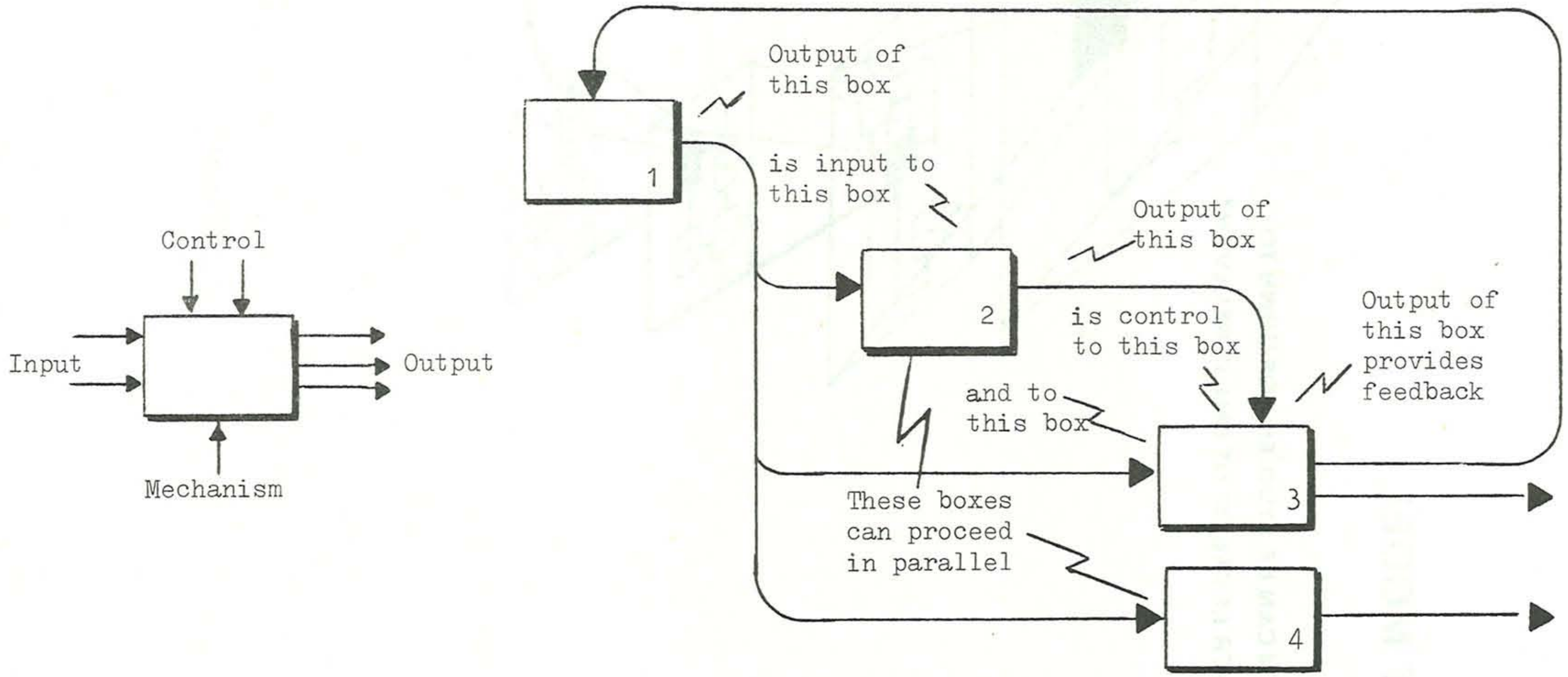
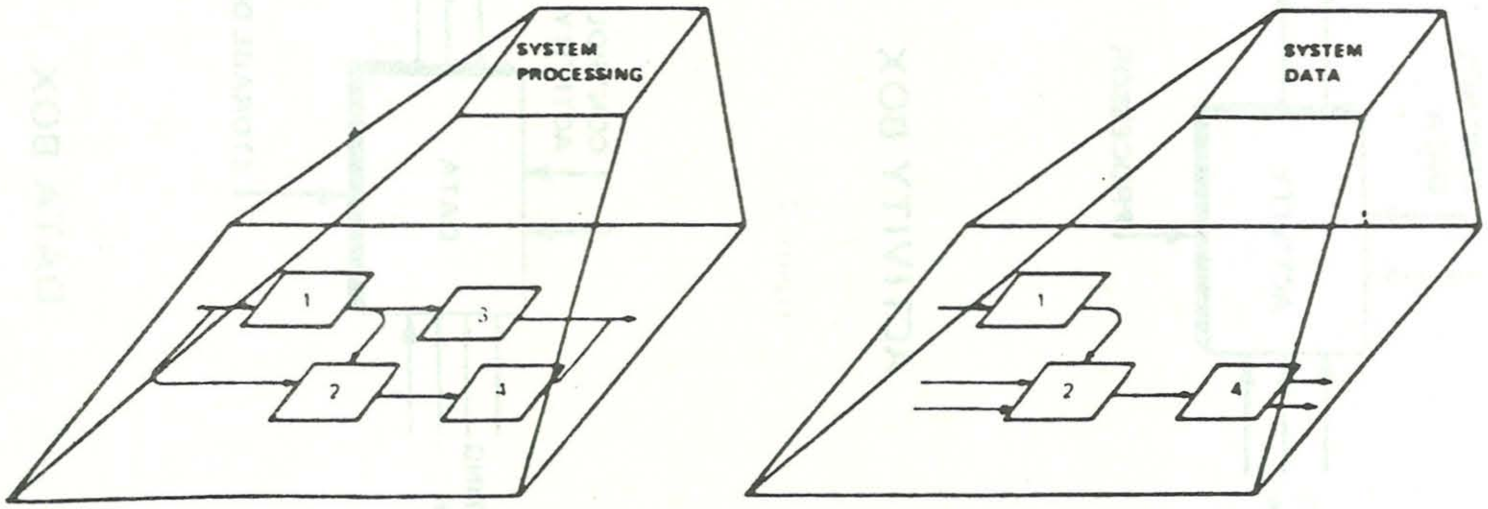


Figure 5

PROCESSES AND DATA ARE MODELLED TOP DOWN



MODEL CROSS REFERENCING HELPS ELIMINATE INCONSISTENCIES AND OMISSIONS

Figure 6

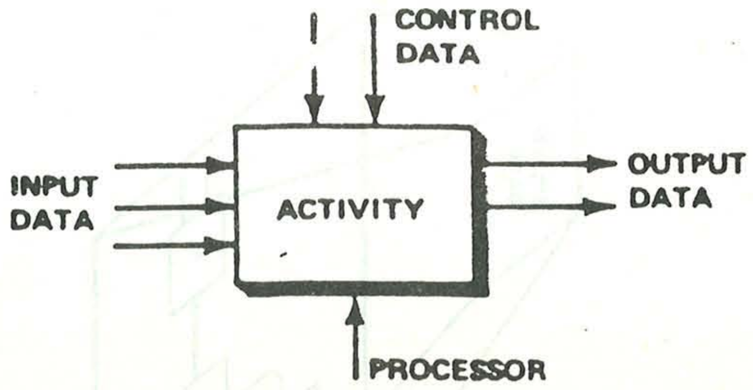


Figure 7

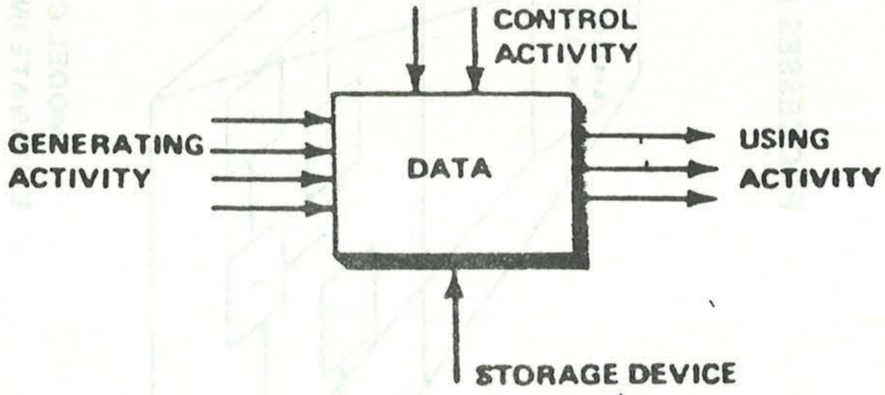
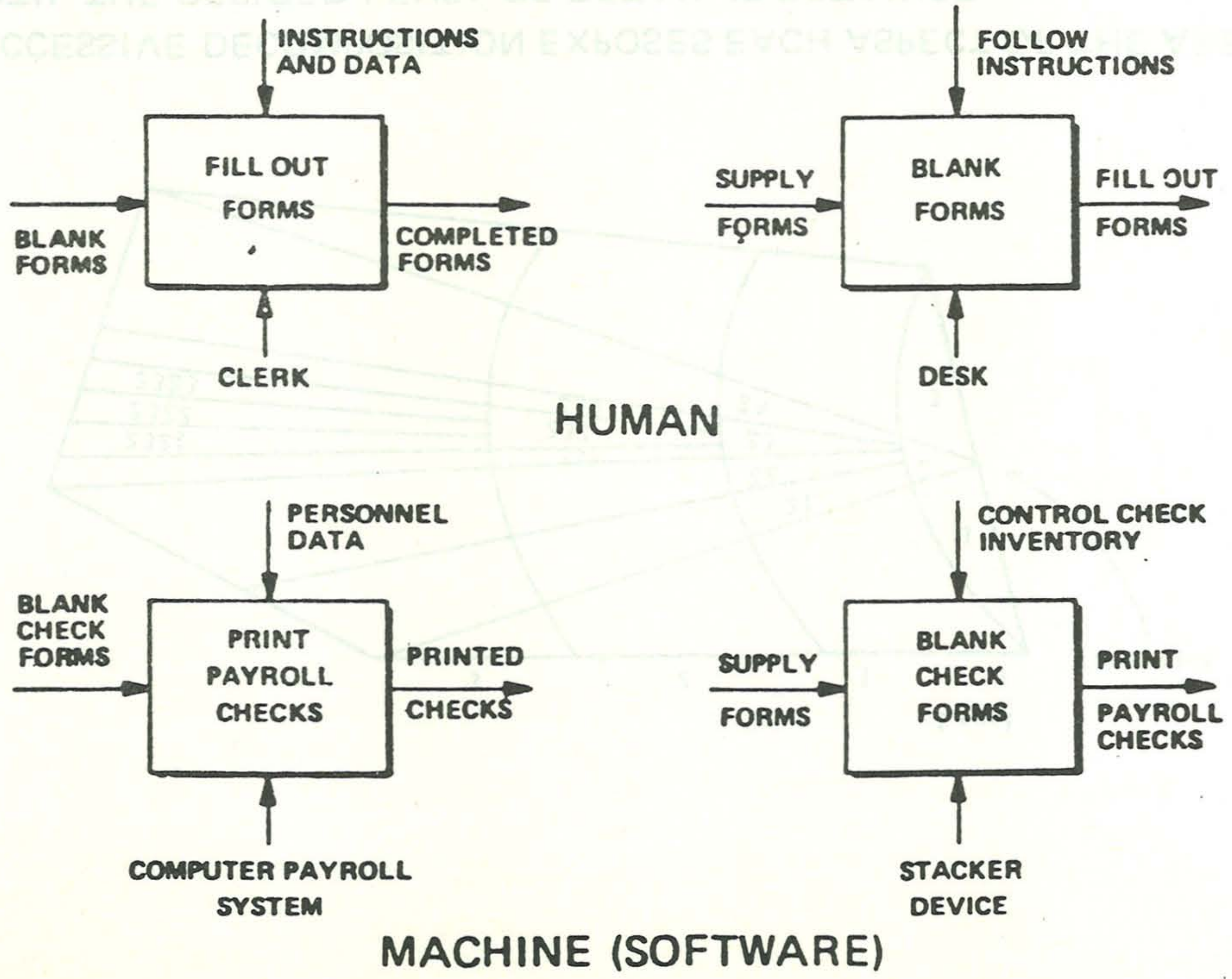


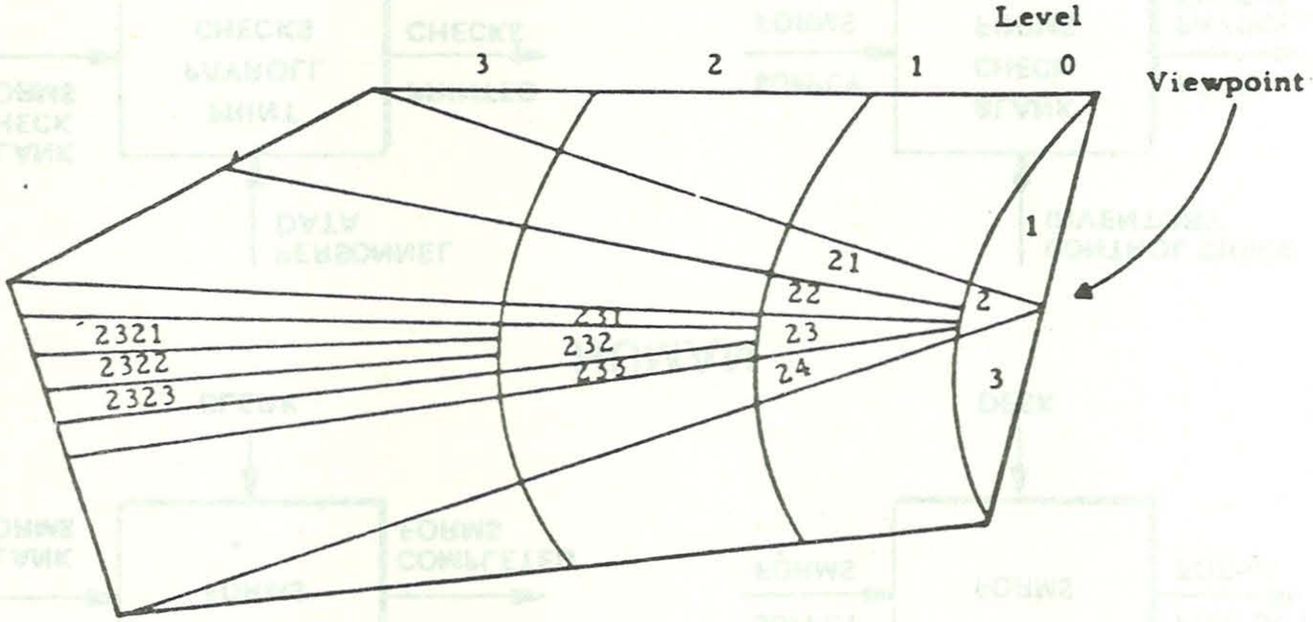
Figure 8

Figure 9

EXAMPLES OF ACTIVITY AND DATA BOXES

CONTRASTING HUMAN AND MACHINE MECHANISMS





SUCCESSIVE DECOMPOSITION EXPOSES EACH ASPECT OF THE AREA UNTIL THE DESIRED LEVEL OF DETAIL IS OBTAINED

Figure 10

ANALYZING A TOPIC FROM MULTIPLE VIEWPOINTS

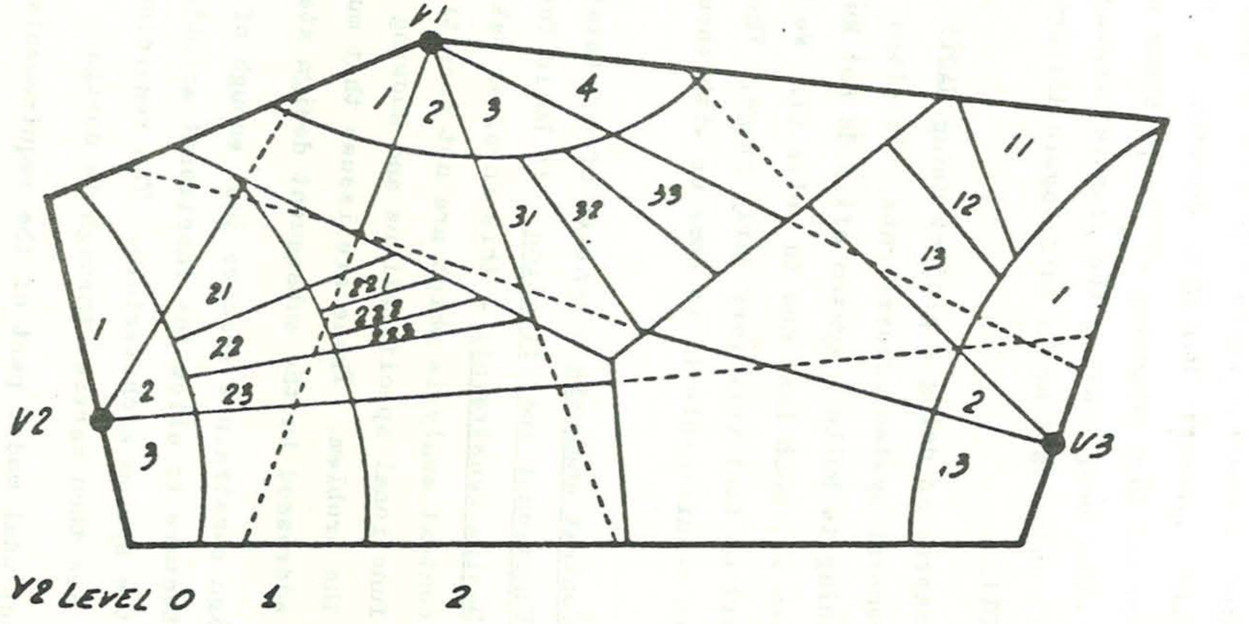


Figure 11

Part 2

In this second lecture I would like to talk about the basic steps of this SADT approach. Our main objective is to organise our own thinking, our own understanding of the problems and then to convey that to other people using the graphic language of SADT and to receive their understanding in turn (again through the graphic language of SADT).

So what happens in doing a project using SADT? We start off by studying the general system requirements. It often turns out that the people wishing to build a system often do not know precisely what the problem is, much less how to solve it. We must, therefore, be able to start at that very very early stage. The purpose of studying system requirements is to come up with three kinds of information:

(i) Context analysis - context of the total set of requirements.

(ii) Functional specification - to build functional models.

(iii) Design constraints - this involves taking parts of the context analysis which are not directly reflected in the functional specifications and showing how they relate to the problem. These are issues that must be further addressed in the subsequent design stages.

The Design constraints require just enough of a sketch of a system architecture to allow the functional architecture to be mapped into the system architecture. The remaining parts of the requirements are then carried through to design.

The functional model part of the requirements' definition study guides the selection of other viewpoints in design, which model how to realise a system that will have a functional behaviour satisfying the requirements. In this process you identify several different viewpoints all of which are treated as the mechanisms (the bottom part of the boxes) which can then be cross related.

Just to put this in graphic terms, let us say I have made such a top-down decomposition where I have broken the total picture into four parts (Figure 12) and broken these parts into further parts as shown for view X. When I take a different viewpoint (view Y) and decompose it, how do I obtain a sharing between the two models - saying that a particular box of one model is intended to be the same as some box of the second model? This is done by cross linking and is made possible by mechanisms. To summarise then - we break the system as a whole into a limited number of modules which are connected together by interfaces. The arrows of Structured Analysis diagrams (input, output, and control) indicate how they are connected together. This step is repeated for each module in turn until the system has been described in sufficient detail.

When looking at a system top-down, which aspect should we model first? If I realise that we are going to have different viewpoints, how do I decide where to begin? There is no single answer to this because some sets of requirements (problem areas) will call for systems that are best thought of, to begin with, in terms of data modelling - you are more interested in the kinds of things that are in them and make them up, whereas some sets of requirements will call for systems where you are more interested in behaviour - what people/processes/devices will do with different things. The chart of Figure 13 shows the main steps of studying system requirements. Not listed on this chart is one point which we find very important. This is that not only do we make models of the problem, but we also make models of the project. When you are using SADT to solve problems, you almost always start out by making a <u>project model</u>: (i) who is on the project; (ii) what roles do they play; (iii) what budgets, schedules etc. By modelling the project, you are able, within that context, to make assignments for the orderly processing of the things that have to do with the problem itself.

In order for a team of people to work together we first of all have to decide on what tasks each one is to perform and where do these tasks overlap? What are the roles and responsibilities? We have to take the product of their work and make it come out as a team effort and not just a pile or results. So the key thing is to have a common language for communicating and to keep precise records of what occurs in the project and above all to maintain visibility at every stage, that is, it has to be easily readable. I shall now describe the main aspects of the SADT graphic language.

There are three basic stages that go into the application of graphic modelling language itself (Figure 14): (i) interviewing, (ii) drawing, and (iii) reviewing. When people are trained in the SADT method they are taught how to go out and absorb information in fields in which they have only a peripheral knowledge. They are trained not to let their own viewpoints intrude; they are trained not to ask leading question (like lawyers). They are even taught a few things about behavioural psychology, for example when to keep quiet, when to say something. The aim is to come as close as possible to the problem, even though the person to whom you are talking has trouble in describing it. Then comes the drawing of the diagrams. Most important is the reviewing. Reviewing takes place all the time. In fact for interviewing, what you do is to come back to the same source with your efforts - diagrams - so that the person from whom you got the information can check that you got it right. The control and support function of Figure 14 corresponds to the program librarian in software projects.

I shall now describe the author-reader cycle (Figure 15). An author first draws a set of diagrams (and sometimes there is a structured text accompanying the diagrams). He makes a reader kit and gives it to one or more readers. The readers are not only taught how to read a diagram, they are also taught, if they are commentators, how to ask questions they have about a diagram. The reader kit, with the reader's comments and reactions, comes back to the author. The author now writes, on the same copy, his reaction to the reader's comments, and sends it back to the reader. Only if the reader and author have open questions do they ever get to meet. The key thing is that because the reading is always top-down, the context of the intended communication is always very closely bounded. This means that these comments on the diagrams can be very very short and still communicate precisely.

The Structured Analysis diagram form, has various boxes across the top that are filled in for control information (author, date, project etc.) and included is a row of numbers (1 to 10). These numbers are used to record the sequence in which reader's comments were made. This is very helpful as the history of the entire construction process is available. The chart of Figure 16 gives the time guidelines. About two to five diagrams a day can be produced. It is much like producing modules in a high-level language. Figure 17 shows how the diagrams are published and bound together. The reading rule is 'only read the text last'. The first thing you do is to read through the names of boxes. Now in what context am I to understand those things? That is the context of the parent - which is represented by the arrows that go off the page or are not connected to anything. With this knowledge you can start looking at the inner arrows and seeing what they mean. Only after this do you read the text which only highlights the main theme. Figure 18 shows how a 'parent box', Figure 18(a), is related to its corresponding child diagram, Figure 19(b). A simple and obvious numbering scheme is employed for numbering the arrows. Finally I will briefly discuss the meaning of double headed arrows, such as in Figure 18(b). Figure 19(a) shows a double headed arrow with a dot at each end. What this means is that box A is a control on box B and conversely, box B is a control on box A. It says that there is cooperation between the boxes, like query and response. The purpose of a dot is not to say that 'run that arrow backwards' but to say 'interchange the relationship of the boxes'. Thus Figure 19(a) is equivalent to Figure 19(b); Figure 20 also shows one such equivalence.

Conclusions

The SADT method has been applied successfully to a wide range of planning, analysis and design problems involving men and machines. The well-structured method of documenting provides a significant aid to all aspects of system design.

Questions

Many people in the audience wanted to know about the training experience for SADT. The speaker replied that he has found that the training of activity modelling was usually easy as compared with the training of data modelling. The most difficult aspect of the training was mechanisms. In software terms, choosing mechanisms is equivalent to choosing levels of abstractions. Professor Turski said that in Figure 19(a) the left pointing and dotted arrow touches the output side while in Figure 19(b) it touches the control side, thus destroying the symmetry. The speaker replied that the 'double headed arrows' is a notation saying, for Figure 19(a), that both A and B are related output to control. It is incorrect to say that there is a symmetry. There is, in fact, asymmetry in Figure 19(a) or in Figure 20. Figure 19(a) says that box A dominates box B, even though they have two-way relationship. The dominance is directed by the output side. Consider a more complicated example of Figure 21. Here, A dominates B with respect to the X/Y relationship, for example, A sends a query to which B must respond. Also B dominates A with respect to the W/V relationship. Professor Seegmüller asked whether the dominance is indicated by the boxes that are 'up'. The speaker replied that it is not so, but the staircase method is the preferred way of drawing. The dominance is directed by the output side.

SADT MODEL

MULTIPLE MODELS ARE REQUIRED TO REPRESENT THE MULTIPLE VIEWPOINTS OF REQUIREMENTS DEFINITION

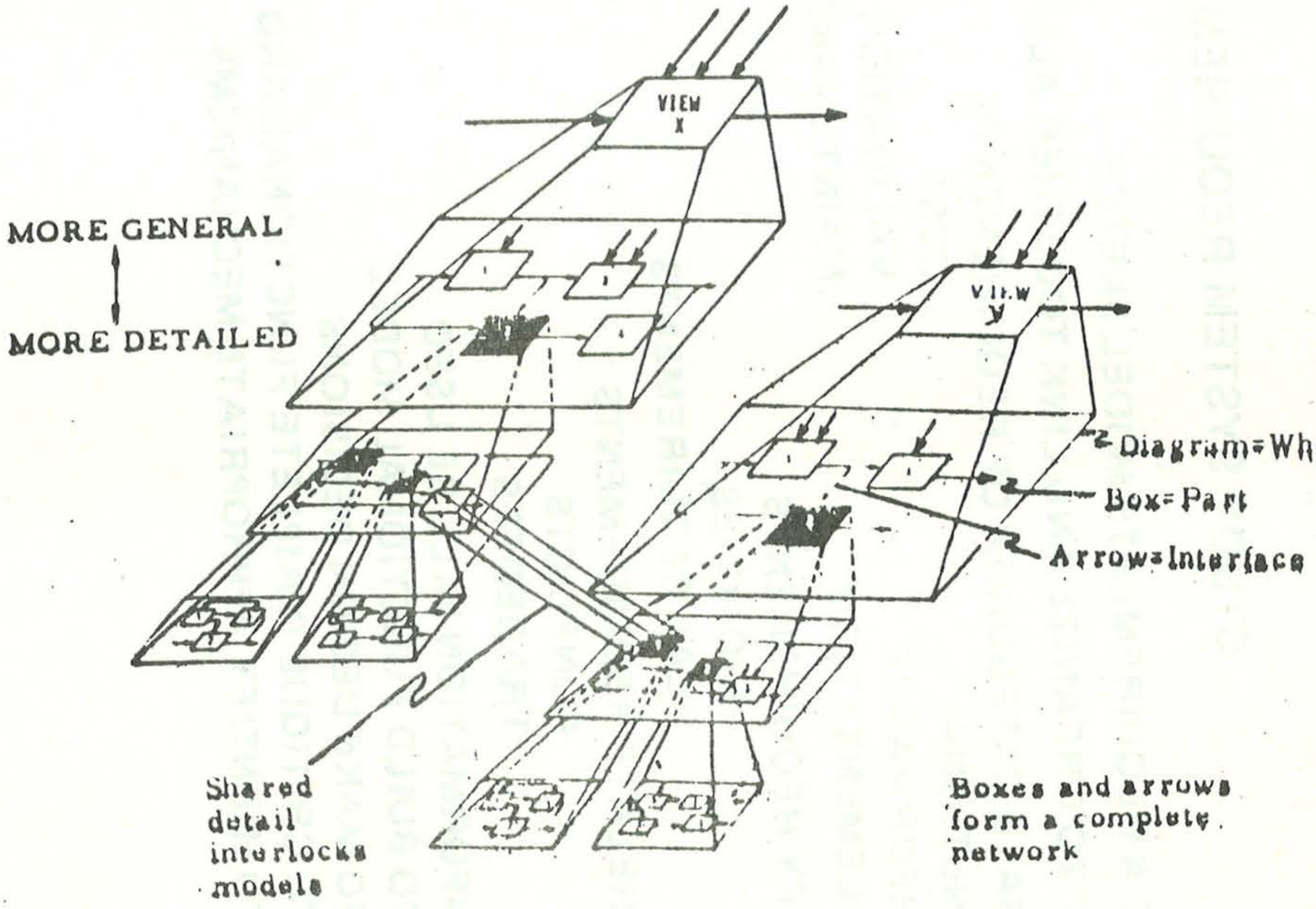


Figure 12

STUDY SYSTEM REQUIREMENTS

MAKE A REQUIREMENTS MODEL SKETCH

USE IT TO ORGANIZE AND LINK TOGETHER ALL REQUIREMENTS DOCUMENTS

SEPARATE CATEGORIES OF REQUIREMENTS

FUNCTIONAL	COST
PERFORMANCE	VALIDATION
IMPLEMENTATION	MAINTENANCE

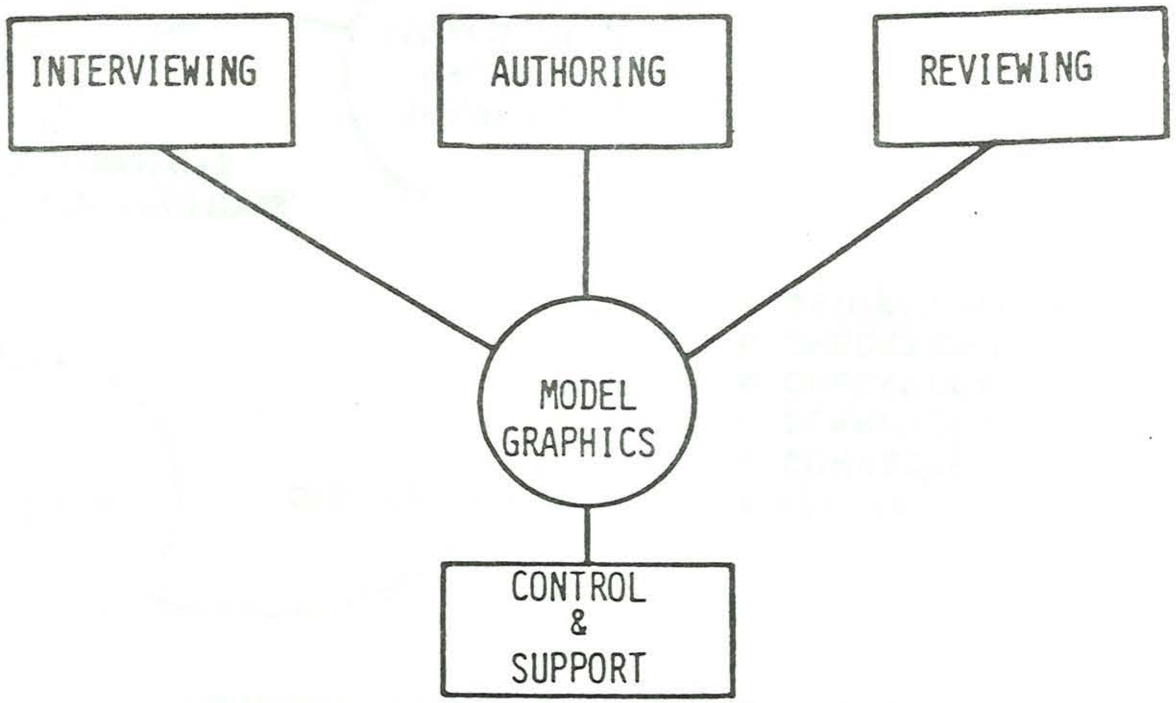
CLARIFY REQUIREMENTS
- PRIORITIES
- REAL REQUIREMENTS

INTERRELATE REQUIREMENTS
- CONFLICTS
- TRADEOFFS

THIS INFORMATION WILL BE USED
- TO BUILD FUNCTIONAL MODEL
- TO MAKE DESIGN DECISIONS
- TO CRITIQUE COMPLETE FUNCTIONAL AND DESIGN MODELS
- TO IDENTIFY APPROPRIATE MECHANISMS

Figure 13

THE SADT METHODOLOGY

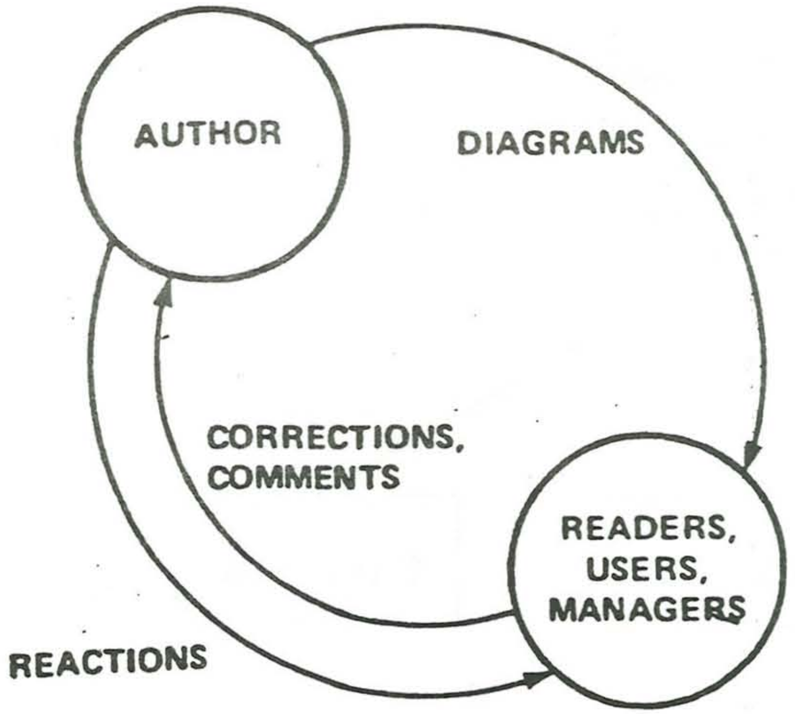


A COMPLETE ANALYSIS AND DESIGN DISCIPLINE
WITH FULL EDUCATION AND INSTALLATION SUPPORT

Figure 14

SADT AUTHOR-READER CYCLE

CONTINUOUS PEER REVIEW OF ALL WORK TO ENSURE QUALITY AND REDUCE ERRORS



A READER
- CORRECTS SYNTAX
- CLARIFIES TERMINOLOGY
- CHECKS CONSISTENCY
- CHECKS COMPLETENESS
- ASSURES ACCURACY

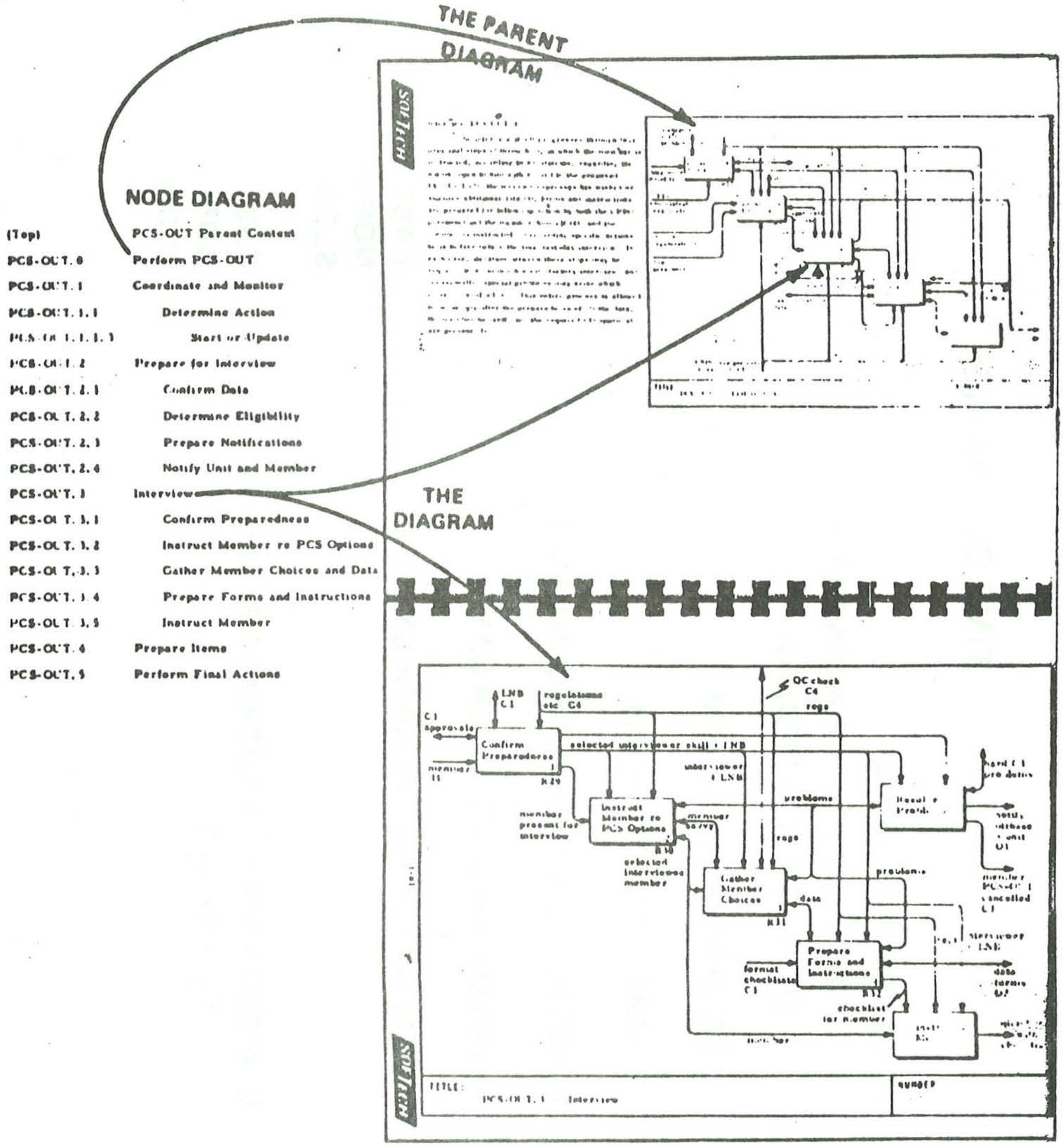
Figure 15

TIME GUIDELINES

TO DECOMPOSE A BOX		15 MINUTES
FIRST VERSION OF A DIAGRAM		60 MINUTES
CRITICIZE AND REDRAW		10-30 MINUTES
TO READ A NEW DIAGRAM		10-40 MINUTES
TO READ A NEW VERSION		5-20 MINUTES
TO REACT TO A READER'S COMMENTS		5-20 MINUTES
DIAGRAMS PRODUCED PER DAY		2-5
ELAPSED TIME FOR READER CYCLE ON A KIT	FAST	4 HOURS
	NORMAL	1 DAY
	SLOW	2 DAYS
ELAPSED TIME FOR AUTHOR REACTION TO A READER	FAST	2 HOURS
	NORMAL	4 HOURS
	SLOW	1 DAY

Figure 16

SADT(TM) ORGANIZES DOCUMENTATION FOR TOP-DOWN READING



NODE DIAGRAM

(Top)	PCS-OUT Parent Content
PCS-OUT.0	Perform PCS-OUT
PCS-OUT.1	Coordinate and Monitor
PCS-OUT.1.1	Determine Action
PCS-OUT.1.1.1	Start or Update
PCS-OUT.2	Prepare for Interview
PCS-OUT.2.1	Confirm Data
PCS-OUT.2.2	Determine Eligibility
PCS-OUT.2.3	Prepare Notifications
PCS-OUT.2.4	Notify Unit and Member
PCS-OUT.3	Interview
PCS-OUT.3.1	Confirm Preparedness
PCS-OUT.3.2	Instruct Member re PCS Options
PCS-OUT.3.3	Gather Member Choices and Data
PCS-OUT.3.4	Prepare Forms and Instructions
PCS-OUT.3.5	Instruct Member
PCS-OUT.4	Prepare Items
PCS-OUT.5	Perform Final Actions

DOCUMENTATION IS PRODUCED CONCURRENTLY WITH DEVELOPMENT

2-68

Figure 17

I N T E R F A C E   M A T C H I N G

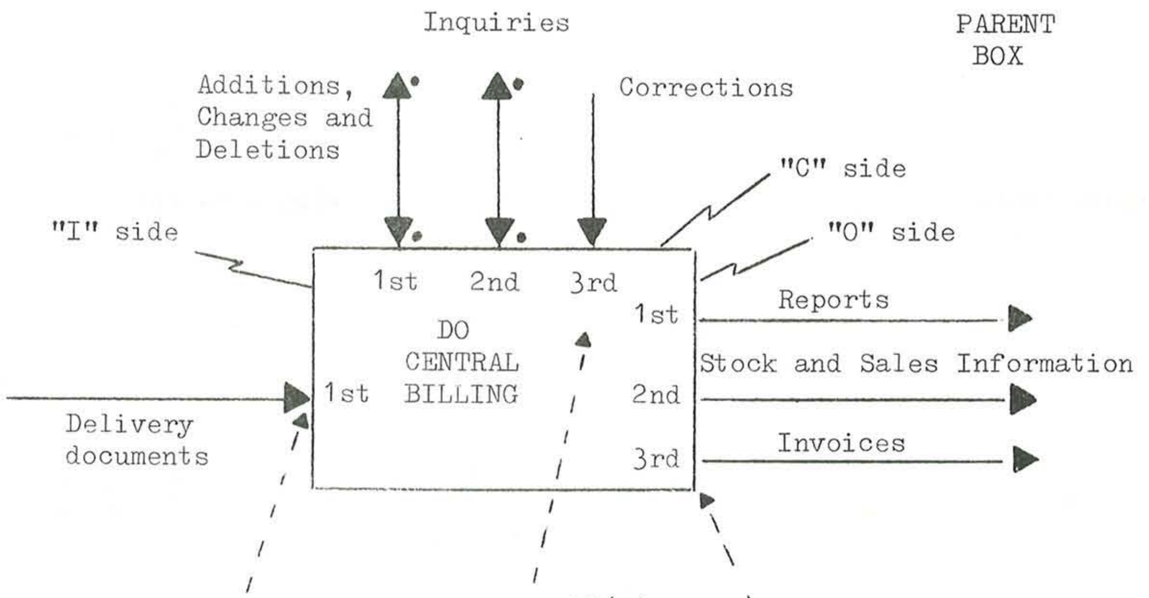


Figure 18(a)

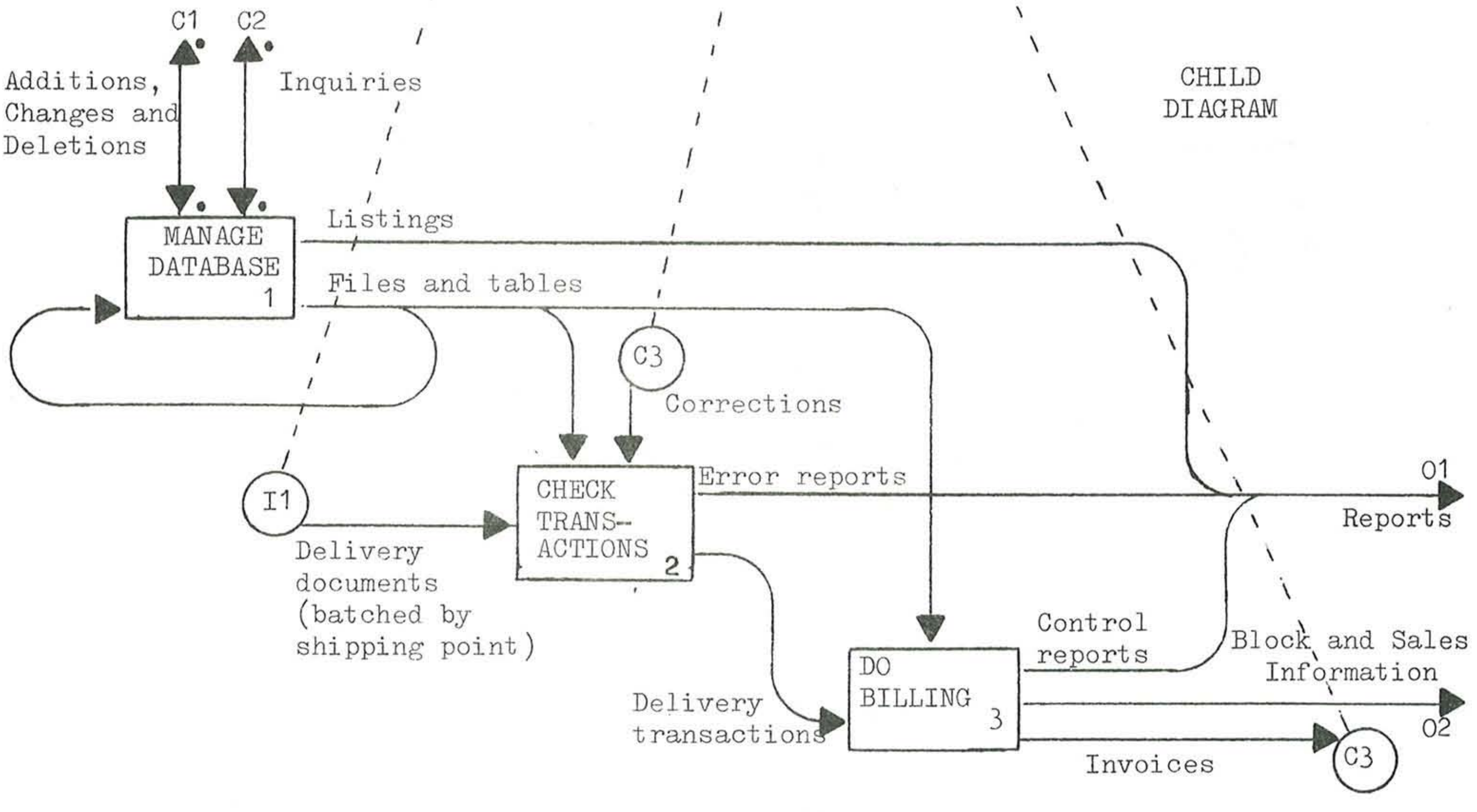


Figure 18(b)

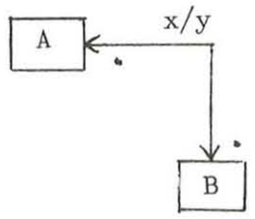


Figure 19 (a)

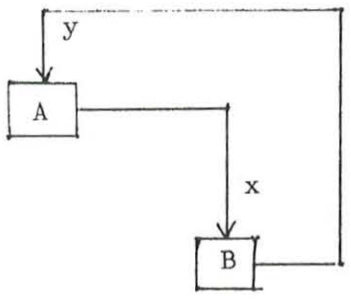


Figure 19 (b)

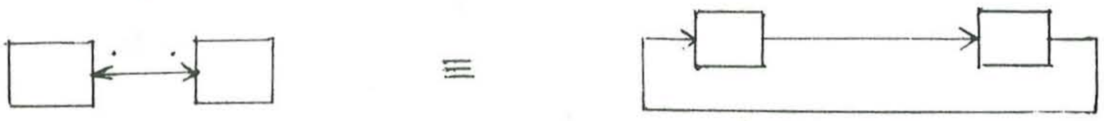


Figure 20

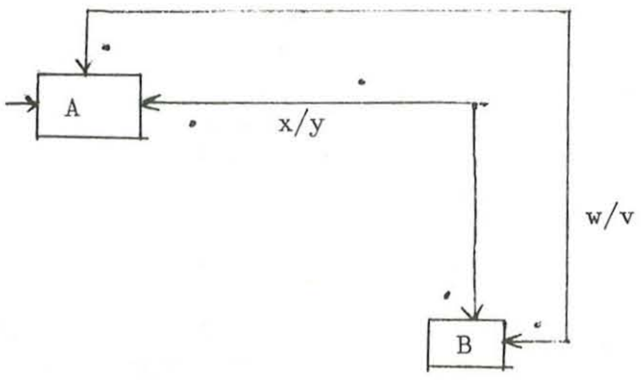


Figure 21

Part 3

The subject matter of the model that Mr. Ross looked at was concerned with sharing courses that different parts of the armed services had developed for training people to do various tasks. The basic idea being that there are enough similarities, and certainly enough investments made in documenting systems for teaching, that sharing would be advantageous. Having said this Mr. Ross went on to describe how Structured Analysis could be used to design such a system.

The setting of the stage for a model would normally be a verbal description of the basic purpose and objective of the subject, which would be followed by the topmost level diagram, which is not a proper Structured Analysis diagram, because one is only supposed to be getting the boundary conditions of what would be a parent box on a higher level diagram if it were to be drawn.

Figure 22 is the topmost diagram for the system outlined above, and indicated some of the features of the diagram that he had not mentioned in his previous talks. These were 1) the 'detailed reference expression', in this case the page number 5-29, which says where the diagram of a full exposition of what the box represents may be found and 2) the Author number (on this diagram, which is for publication, this is replaced by the page-number) which is used to indicate the author of the diagram and its creation date.

Figure 23, shows a Structured Analysis text, and pointed out that it is a high-level description with coded references embedded in it to indicate very precisely which point, arrow, or pathway is being referred to. The text displayed was not a very detailed one as it was for the highest level of the system, however as one gets further into the system the detail increases and box numbers are put on the front of reference codes, thus 302 refers to output two of box 3, so using this, and in conjunction with 'and' and 'or,' the direct cross referencing between text and diagrams is just as precise as one could wish. In fact, a whole diagram can be represented by a stream of single characters.

Mr. Ross then went on to describe how information develops as one reads a Structured Analysis diagram by working through the system described above.

The first thing a reader must do is to examine the contents of the boxes so as to discover the general breakdown of the subject matter, and this serves to orient the reader's mind. Therefore, referring to Figure 24, one sees that, whatever course descriptions and units are, they are going to be screened and the unit descriptions are going to be used to maintain a data base. Also, course descriptions and units have to be found in the data base, and their distribution indicated and assured. In Figure 24 the instruction to assure distribution was probably added in the reader cycle of the design development, when the author put his document out for suggestions. For this, what they consider to be primary flows, are made heavy in the diagrams, the others just serving a supporting role.

The next step in reading the diagram is to determine the positions of the boundary conditions, which are the interface conditions already seen on the parent box. For example, in Figure 24 the actual material descriptions are coming in to be screened and distributed. It is advisable to have both the parent and the child diagrams in front of one. In fact to read a model properly one ought to stick the diagrams on a wall in a tree structure and scan back and forth to get benefit from having everything in front of one at once.

Then the reader should proceed to examine the distribution of controls and functions in the diagram. Thus, in Figure 24 the screening process takes the materials presented, following the rules, policies and procedures, to come up with new and altered unit descriptions, being what are maintained in the data base. Whenever a request comes in one extracts from the data base what is needed and answers that request, but at the same time one is updating what was transmitted, so that the actual distribution of the material is just an inquiry into the system.

Next, proceed one more stage down the model, (Figure 25). This process is split roughly into these pieces: 'What are the selection rules?', 'Select on basis of these rules', 'Review unit descriptions' and 'Make either new or altered descriptions to be added to data base', (the ultimate disposition of everything being to come out and go into the data base). This diagram, illustrates use of mechanism arrows showing that training command HQ, HQ personnel and Traidex employees will have a role in setting the course selection rules. However once the rules are set, the selection of courses and their entry into the data base depends on the people who developed the course. The feedback arrow from 2 to 1 in Figure 24 shows that by maintaining records of usage the course creators can influence how the rules are set up.

Moving down yet another stage (Figure 26), Mr. Ross indicated that the layout of the information (staircase fashion left to right) helped greatly to increase understanding, even at the fine level of detail that we are now at. In the diagram we see that the only role that training command HQ has is to say what they want applied, which could not be deduced from the parent box, which merely said that the training command HQ, the Traidex Centre personnel, and the course organiser will all participate in setting the rules. That, of course, may be all that somebody needs to know. 'As long as I'm represented I'm happy', but whether one is really happy depends on what happens when one looks at the next level and sees how the actual roles are spread. Notice, for example, that the Traidex central office is playing quite a dominant role in interpreting how all the functions are performed here.

At this point Mr. Ross told the audience that the level of information provided by the diagram he had been showing was about average, some people being better at producing small, precise expressions for things. The diagrams produced by these people are a little more open, and in fact this is how one wishes them to be, as too much clutter almost always causes confusion. Here <u>Professor Kerner</u> asked if the staircase layout of boxes was always used, or whether it was sometimes necessary to break from it. Mr. Ross answered that the staircase layout was not rigidly enforced, but that from the point of view of ease of reading it was definitely advantageous, as it left the bottom of the diagram for details of mechanisms and the top for feedback and correction paths, and it seems to follow the idea of dominance and sequencing which eases the construction of activity diagrams especially; of course if any special symmetries arose in diagram these would be emphasised, even if they were not in staircase form, Mr. Ross added.

<u>Professor Pyle</u> then asked whether an arrow which is labelled before a split implies that the same information is going to two different places. The answer to this was yes, but only to the precision that this level of diagram requires. In other words it may be that when one gets to the detailing, more layers down, one finds that not all the information is used in both places, but it would clutter one's mind to have more facts at this point. <u>Professor Randell</u> enquired if there was a mechanism for selection of courses not described in the current diagram or its parent. Mr. Ross replied that if box A12 was detailed one would find, in selection of courses, that both the selected and non-selected would be reported back up and one can see in the detailing of the current box where the specific split happens.

In general, what happens in a model or collection of models is that if one looked at the finest level of detailing there would be many tips of roots that come together, join, and make a trunk which then penetrates through various higher levels, and branches out into various leaves. The whole purpose of the disciplined thinking and disciplined use of the notation is to simplify the whole, obtain proper terms for everything, and put the boundaries in the right places, to the end of enabling every reader to understand different parts of the structure in such a way that his mind will somehow put it all together in his own way and make sense of it.

Professor Pyle asked a question at this point about the relationship between a diagram and its related data diagram, to which Mr. Ross replied that though the data has mechanically the same form, there is much greater variety, because people do not yet feel happy with data modelling, because they find it hard to abstract real things, and that is what must be done to have hierarchic top-down data modelling. One has to talk about kinds of things but not arbitrary kinds of things, they have to be kinds of things organised functionally to serve the purpose that one has in mind, and that, Mr. Ross maintained, usually requires a new sort of insight. Most people solve problems by going very specifically and rapidly into some detail of the problem, to find the chink in the armour and start tearing it apart, whereas, when doing design or analysis by these higher level methods one must do exactly the opposite.

Professor Pyle now inquired whether the data diagram could be deduced from the activity diagram of a system; the answer was 'no'. He then asked why there were no data diagram cross reference numbers on the activity diagrams that Mr. Ross had been displaying. This was because they had not been tied, which would entail labelling each arrow in both types of diagram (arrows in A diagram are D flows and vice versa), which provides consistency checks on the flows.

At this point Mr. Jackson wanted to know if there are the same number of boxes in the D diagram and the A diagram, to which Mr. Ross replied that this was not the case, because the method of classification is to take an arrow segment in one of the models and classify it down until it tries to split in the other model; it could happen that they all end up in high level boxes and the lower level boxes have none in them at all.

Mr. Ross then went on to describe how two different models might inter-relate with each other, indicating that a box could have no detailing in one model, because its detailing is the same as that in another model, so that the two models in fact have a common part and are thus linked together. (Mr. Ross said that this showed that Structured Analysis could be believed, as it would be completely unbelievable if someone claimed that the whole world is built of nothing but trees!) To see how this sharing is achieved, recall that the top three sides of a box - the input, control, and output - are true interfaces, where in an activity diagram, the data arrows make connections from activity to activity. But the bottom side serves a different purpose. If you take any sub-tree out of a tree and it serves the same roles in several places (in each case as the details of output and controls in its own context). This is indicated by an upward arrow called supply which makes that tree accessible as a support mechanism that can be called from each box. This functions exactly like subroutine call with parameters. Each particular call on a support box can be tailored to suit its use, needless inputs, output or controls being marked with an 'X' to indicate their redundancy. A call is indicated by putting a downward arrow in front of the called box designator, which can be as detailed as one wishes.

One important thing about this system is that the mechanism arrows can cut through any number of onion skin layers of nesting, and can collect together and branch too, which means that models can share as finely as required.

Mr. Jackson asked if this was just a convenient shorthand, to which Mr. Ross replied that it was not *just* a shorthand, but could be used, for example with conditional reference expressions to direct people to different realisations of a given box, depending on their interests, which cannot be done with just nesting.

Professor Randell then said that he tended to think of a Structured Analysis diagram as being one step away from the soldering iron or representing an actual simulation, and asked to what extent this was and could be the correct view, and if so how much had they been used this way. Mr. Ross replied that it could be a correct view and the diagrams had been used that way, but that there were no actual simulators ready to run, though algorithms to drive GPSS had been worked out.

At this point Mr. Ross answered an earlier question from Professor Lehman, who had asked how one could ensure that everything at the lower level in a diagram properly represented that at a higher level. Mr. Ross did this by defining two boundaries for a diagram, one, outer boundary, which is defined by what is shown on the A-0 diagram, and the other, the inner boundary, which is defined by the set of boundaries of all the finest-level boxes together. Then if these two boundaries are identical, the lower levels truly reflect the upper levels. With this Mr. Ross finished his lecture.

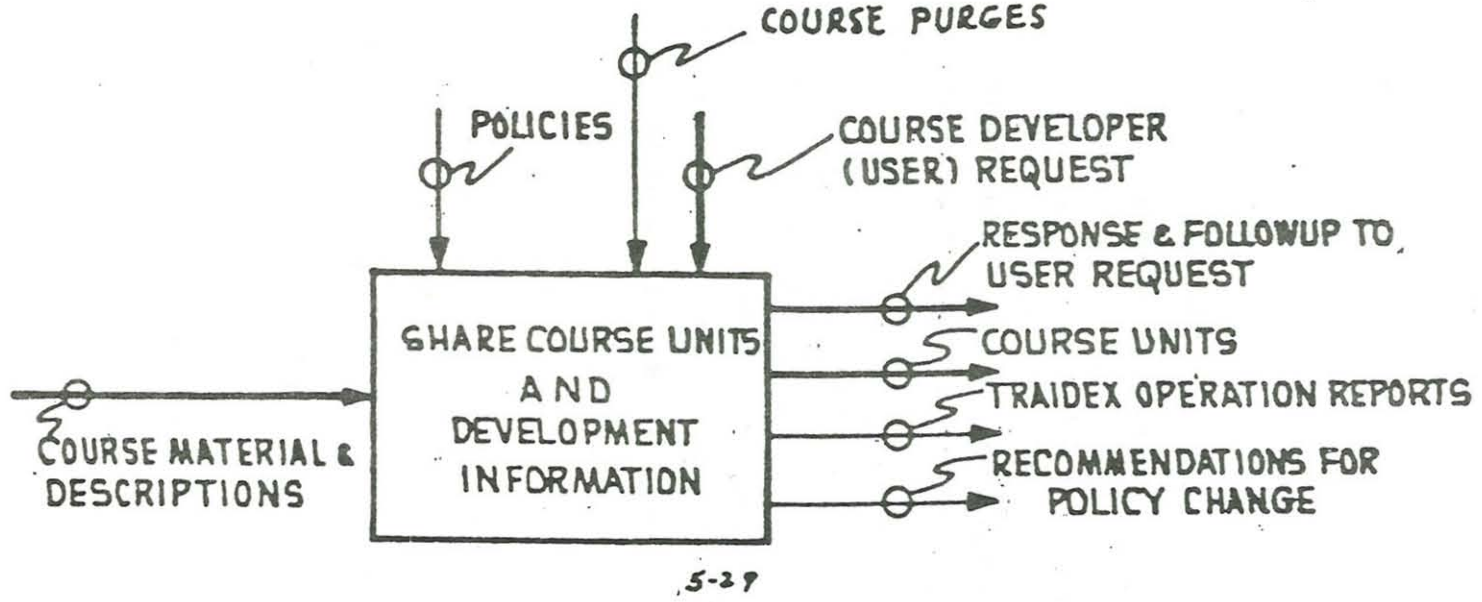

COURSE PURGES
POLICIES
COURSE DEVELOPER (USER) REQUEST
RESPONSE & FOLLOWUP TO USER REQUEST
COURSE UNITS
TRAIDEX OPERATION REPORTS
RECOMMENDATIONS FOR POLICY CHANGE
SHARE COURSE UNITS AND DEVELOPMENT INFORMATION
COURSE MATERIAL & DESCRIPTIONS
5-29

Figure 22

Figure 23

A-0T. SHARE COURSE UNITS AND DEVELOPMENT INFORMATION

The scope of the TRAIDEX system design model is bounded by the input of course material and descriptions (I1), produced in the various service schools and other course development sites, and delivered as outputs (O2) to other course developers who wish to review the course units in depth in order to decide whether reuse is possible. This process is controlled by policies (C1) from the ITRB and the Training Commands, by directives from these groups and others to purge obsolete course material (C2), and by the Course Developer's request (C3) for material that he expects to reuse in his work. In order to assist the requesting user in his attempt to locate learning materials suitable for reuse, the system provides responses (O1) to his requests which allow him to examine descriptive material about selected course units and to make the final decisions on whether or not to request copies of specific learning materials. The system also provides information about its own operations (O3) and can make suggestions regarding beneficial changes in its operating policies (O4).

Note that the most important input, output, and control arrows have been emphasized by making them heavier than their companions.

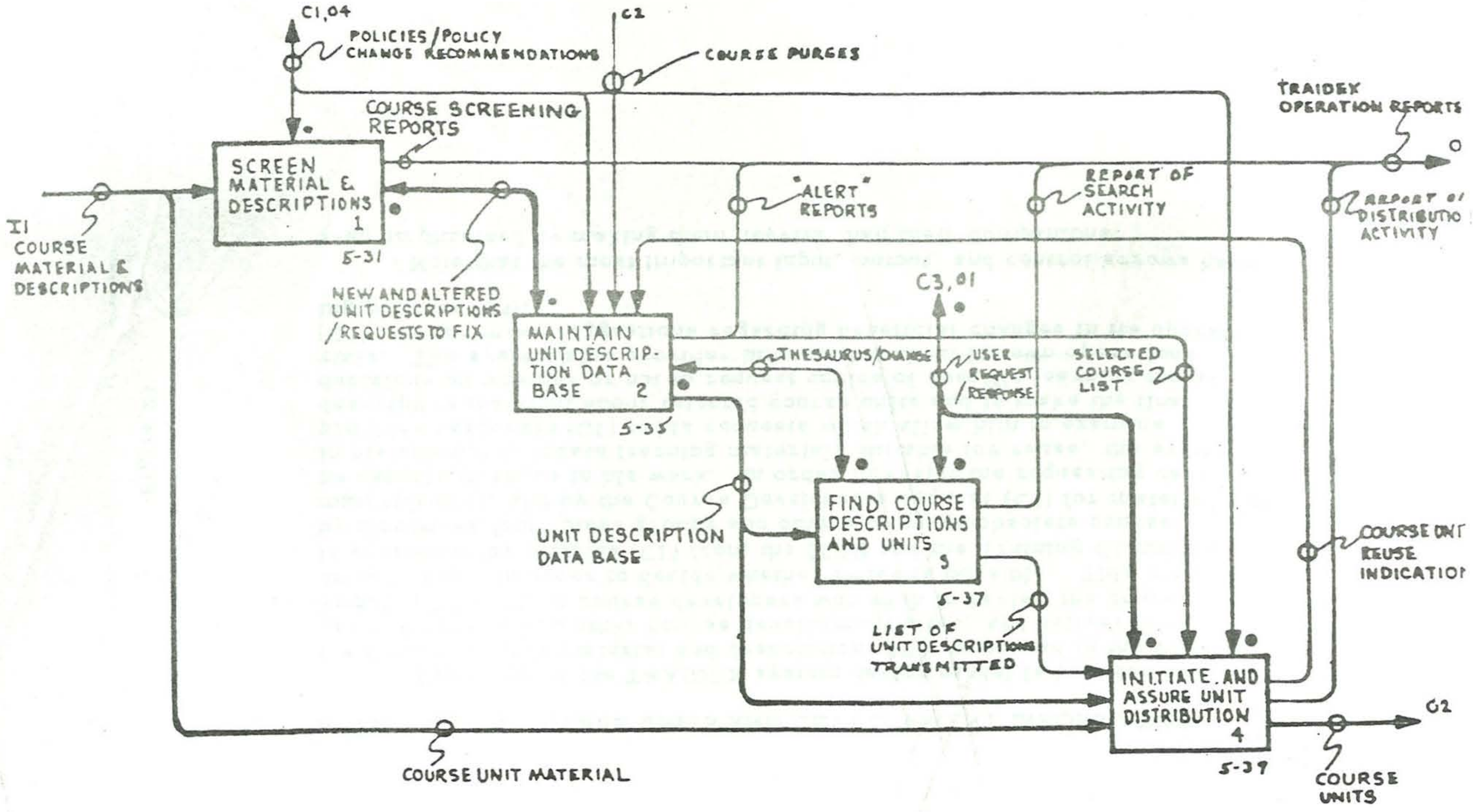

C1,04
POLICIES/POLICY CHANGE RECOMMENDATIONS
C2
COURSE PURGES
COURSE SCREENING REPORTS
TRAIDEX OPERATION REPORTS
O
SCREEN MATERIAL & DESCRIPTIONS 1
5-31
"ALERT" REPORTS
REPORT OF SEARCH ACTIVITY
REPORT OF DISTRIBUTIO
ACTIVITY
I1
COURSE MATERIAL & DESCRIPTIONS
NEW AND ALTERED UNIT DESCRIPTIONS /REQUESTS TO FIX
C3,01
MAINTAIN UNIT DESCRIPTION DATA BASE 2
5-35
THESAURUS/CHANGE
USER REQUEST /RESPONSE
SELECTED COURSE LIST
FIND COURSE DESCRIPTIONS AND UNITS 3
5-37
UNIT DESCRIPTION DATA BASE
COURSE UNIT REUSE INDICATIOI
LIST OF UNIT DESCRIPTIONS TRANSMITTED
INITIATE AND ASSURE UNIT DISTRIBUTION 4
5-39
O2
COURSE UNITS
COURSE UNIT MATERIAL

Figure 24

226

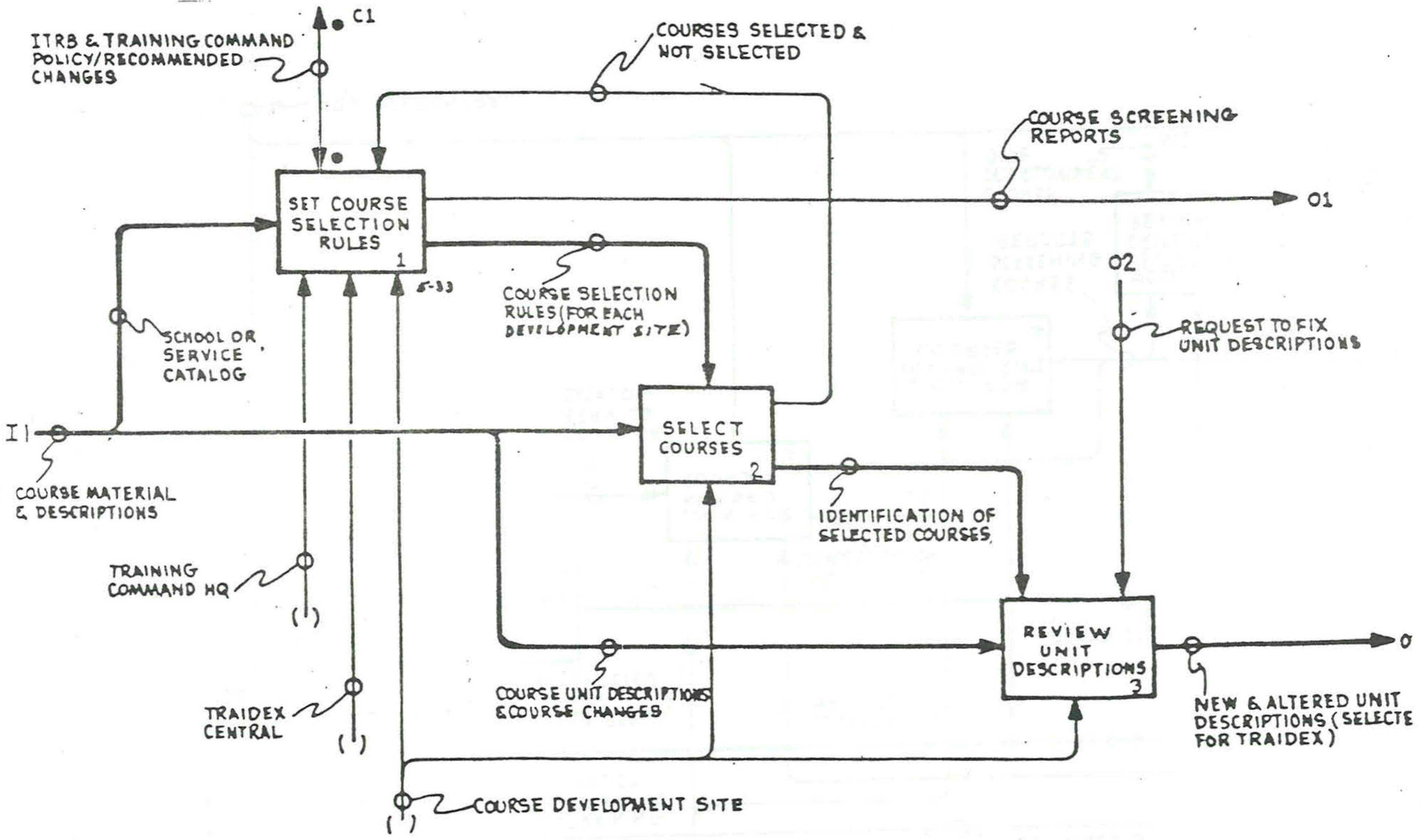

C1
ITRB & TRAINING COMMAND POLICY/RECOMMENDED CHANGES
COURSES SELECTED & NOT SELECTED
COURSE SCREENING REPORTS
SET COURSE SELECTION RULES
1
5-33
O1
O2
SCHOOL OR SERVICE CATALOG
COURSE SELECTION RULES (FOR EACH DEVELOPMENT SITE)
REQUEST TO FIX UNIT DESCRIPTIONS
I1
SELECT COURSES
2
COURSE MATERIAL & DESCRIPTIONS
IDENTIFICATION OF SELECTED COURSES
TRAINING COMMAND HQ
( )
REVIEW UNIT DESCRIPTIONS
3
O
TRAIDEX CENTRAL
( )
COURSE UNIT DESCRIPTIONS & COURSE CHANGES
NEW & ALTERED UNIT DESCRIPTIONS (SELECTE FOR TRAIDEX)
COURSE DEVELOPMENT SITE
( )

Figure 25

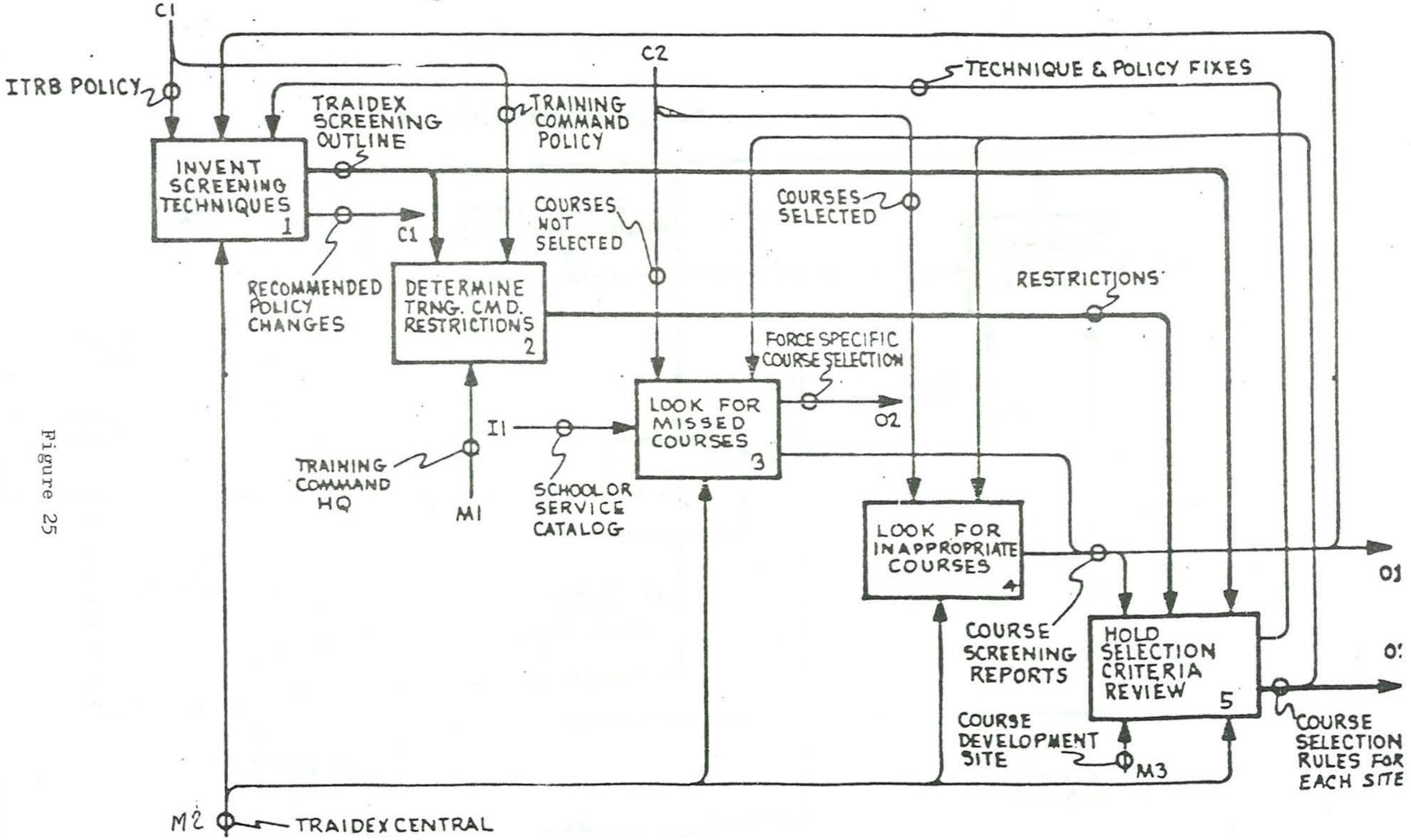

C1
C2
ITRB POLICY
TRAIDEX SCREENING OUTLINE
TRAINING COMMAND POLICY
TECHNIQUE & POLICY FIXES
INVENT SCREENING TECHNIQUES 1
C1
RECOMMENDED POLICY CHANGES
DETERMINE TRNG. CMD. RESTRICTIONS 2
COURSES NOT SELECTED
COURSES SELECTED
RESTRICTIONS
FORCE SPECIFIC COURSE SELECTION
LOOK FOR MISSED COURSES 3
O2
I1
TRAINING COMMAND HQ
M1
SCHOOL OR SERVICE CATALOG
LOOK FOR INAPPROPRIATE COURSES 4
O1
COURSE SCREENING REPORTS
HOLD SELECTION CRITERIA REVIEW 5
COURSE DEVELOPMENT SITE
M3
COURSE SELECTION RULES FOR EACH SITE
M2
TRAIDEX CENTRAL

Figure 25