WORKING NOTE AC69

Alan Creak
16 January 1990

SEMANTICS OF BLOCK DIAGRAMS

At the end of our current work on simulation, we would like to have a collection of software which would understand a picture of an object in sufficient depth to simulate its behaviour over a period of time. It is therefore appropriate to ask questions about the pictures which can be used. For example, we would like to know what sorts of picture can be drawn, whether there are restrictions on the sorts of picture which the software will accept, how the picture is to be interpreted, and so on. This note collects some of my recent thoughts and speculations on this theme.

WHAT SORTS OF PICTURE ?

I have chosen to use the word "*picture*" to emphasise that the restriction to block diagrams (which we shall undoubtedly impose eventually) embodies a decision. Most pictures, other than purely abstract works, go some way towards defining some properties of some system; no picture can define all the properties of the object depicted. We will always need some sort of information *as well as* the picture; the question is how much information we can get out of the picture.

There is a lot of information in a conventional picture, but it is expressed in a syntax which is far from easy to unravel. (It is beguiling to speculate on the problems of simulating systems represented by well known paintings - consider the Mona Lisa, "The persistence of memory", a selection of Escher's works …) The same principle applies to "visual languages". In a rather anecdotal report on a conference[15] we read : "VLs thus far seem to be very low level, and neglect formal semantics, to the exasperation of users". Even if we could manage the interpretation, we would still need lots of additional information : I suspect, in fact, that a general requirement might be to be able to recognise when further information is needed, and to be able to ask for it.

In fact, we shall simplify the problem by restricting the form of our pictures to block diagrams. (This is exactly analogous to restricting written communication with a computer to formal programming languages instead of English : in both cases, we replace a very complex syntax by a much simpler and more regular version.) I think it's useful to bear in mind, though, that we are dealing with an artificially simplified system.

It may be that even with the restriction to block diagrams we are left with a class of pictures too large for our purpose. Are all block diagrams amenable to simulation in some sense ? - or are some forms of diagram associated with systems which could never sensibly be simulated ? I don't think that there's an obvious answer to this question, so I shall defer it until after studying the diagrams.

The next step, then, is to study the syntax and semantics of block diagrams.

BLOCK DIAGRAMS.

A block diagram is a picture in which the objects depicted and the relationships between them are presented as explicitly and as clearly as possible. In a conventional picture, objects are commonly presented quite explicitly (though in more abstract work that may be less true), but relationships are left implicit. In addition, the structure of the object depicted may well be obscured by clutter, or by irrelevant detail - for example, La Gioconda's bone structure is obscured by her rather awkward grimace.

In contrast, the blocks to be seen in a block diagram are abstract representations of objects in the depicted system, and the relationships between them are shown as arcs joining the blocks. In a sense, by defining the sort of picture which the software will accept we have answered, at least in part, two of the original questions. To use block diagrams, though, we still need to ask the remaining one : how is the picture to be interpreted ? We need to define the semantics of the block diagram.

The objects in the diagram are of two kinds, boxes and lines. Each has semantic properties, but clearly the two sets of properties are not independent. I shall begin by investigating the semantics of lines; I assume that the semantics of the boxes are likely to be at least as complicated as that of the lines, as the boxes must reflect the arbitrarily complex behaviour of objects in the real system, while the lines exist only in the abstract diagram of the system and depict connections between things rather than the things themselves.

It could be argued that in that assumption I have artificially restricted the properties of lines - that in fact a line could be a much more complicated thing than I have supposed. I think that this is a matter of how you define the function of a line. I have taken the view that lines should do the least possible consistent with their function of joining things together - that they are not to be seen as active parts of a simulation model, but only as notational devices, with functions analogous to those of the textual notational devices =, ->, <->, etc.. This is not obviously always true of things conventionally, and effectively, represented by lines : electrical conductors may have distributed impedance, a geographical route may alter the properties of objects travelling along it through fuel consumption, wind speed and direction, and so on. I am coming to the view that, because these things are actively affecting the state of the simulation, they should be seen as boxes - though it may well make practical sense when drawing a diagram to depict them as things that look exactly like lines.

Continuing this line of argument leads into strange territory, where boxes and lines become essentially indistinguishable. One is led to wonder whether there really is any difference between boxes and lines. This speculation becomes even more convincing when one considers the hierarchic properties of boxes : we may describe one box by a network of boxes and lines. Can lines also be represented in similar ways ? Why not ? We could represent complicated behaviour in a line by drawing the line as a network of lines and boxes. The two notions converge. It is important, though, that they do converge : that the identity (or near identity) is only seen in complex systems. At the other end of the scale, there are - there must be - elementary lines and boxes. Generally, a line has ends, but need have no active centre; a box has an active centre, but need have no ends. In this note, I shall assume that it is always possible in principle to enclose all active properties in boxes, leaving lines to their simple (or fairly simple) connective function.

Where would we find out about block diagrams ? It seemed reasonable to me that an appropriate place to search would be in works on General Systems Theory, because that discipline purports to concern itself (perhaps[2] among other things) with the behaviour of any systems composed of component parts. I was disappointed in the first book I consulted[1] : whilst the first definition it gives for a general system is sufficiently open to include anything representable in a block diagram, the second in effect insists that all connections to objects must be regarded as inputs or outputs[3]. This restriction was, of course, exactly what I was trying to avoid, and I still see it as an unwarranted limitation of the idea of general systems.

It may be that other works on General Systems Theory would broaden the definition. I haven't had enough time to engage in a protracted search, but I've browsed through two or three promising-looking volumes in the library without much success. An introductory text[4] gives the right sort of definition ("Systems are any set of components which *could* be seen as working together for the overall objective of the whole"[5]), but once again assumes without discussion that all interactions involved inputs and outputs ("Interfaces are those boundaries where two systems meet, such that the output of one system in the input to the other"[6]). A number of impressive block diagrams are presented, but there is no attempt at any clear definition of the meanings of the diagrams. In the later treatment, the author relies strongly on verbal descriptions of interactions between boxes.

A text on simulation[7] proved more fruitful. The importance of *components* and of *component interactions* ("the rules by which components exert influence on each other") are explicitly brought out[8]; but the author shies off precise definition of the sorts of "influence" which can be exerted, relying on case-by-case verbal descriptions of the interactions in the text, and noncommittal "influence diagrams" in illustration. Some cases of systems in which the interactions are not simple input-output links are discussed, though - notably *cell space models*[9], which include finite-element methods and the "Game of Life".

> I found my most enlightening source in an unexpected place, and some time after working through the analysis presented below. It is a book[10] which I have owned for many years, and it was on my shelf of old books, unlikely to be useful again. We all make mistakes. Although, yet again, system behaviour is restricted to input-output interactions[11], these terms are seen primarily in terms of flows of information, and the possible sorts of flow are analysed in some detail[12].

So I've fallen back on guesswork and intuition. That isn't very satisfactory, and I'm open to any suggestions for improvements.

LINES.

What can lines do in block diagrams ? One way to explore the possibilities is to look at some examples. Here is a collection of block diagrams taken (rather naughtily) from the text[1] on general systems theory which disappointed me.

The first example is interesting in several ways. First, the accompanying text states exactly why I expected that general systems theory would be helpful; second, it is a block diagram about block diagrams; third, it has lines which, despite the arrows they bear, cannot reasonably be interpreted as channels of flow for any reasonable commodity. This diagram is, in fact, either a *state diagram*, showing successive stages in the development of a project, or it is a *lattice* embodying a partial ordering of the terms mentioned based on (something like) degree of precision. For a state diagram, the lines represent allowed transitions between blocks; in a lattice, they represent elements of a relation. In both cases, direction is significant.

The principal attractiveness of block diagrams is their simplicity, while their major drawback is a lack of precision. General systems theory models eliminate this drawback by introducing the precision of mathematics, while preserving the advantage, i.e., the simplicity, of block diagrams. The role of general systems theory in systems analysis can be represented by the diagram in Fig. 1.1. General

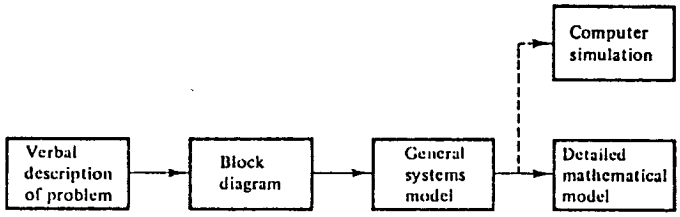

FIG. 1.1

systems models fall between block-diagram representation and a detailed mathematical (or computer) model. For complex systems, in particular, a general systems model might very well represent a necessary step, since the gulf between a block diagram and a detailed model can be too great. The fact that certain general systems techniques and results have become available to treat the systems problem on a general level makes it possible to add this step in practice.

Figure 1.

The next example is unambiguously a *state diagram*, but it differs from the previous example in that the lines are labelled to identify the initial state of the corresponding transition, and the stimulus which caused it. Are these in fact properties of the line ? In accordance with the principle of divesting lines of as much property as possible, I argue that the labels are in fact properties of the box representing the initial state. That box "knows" its own state; and in that box the decision as to which should be the next state must be taken. The lines are therefore just the same as those defined for the previous diagram when considered as a state diagram; perhaps this example emphasises that the lines simply *show the way* to reach the next state.

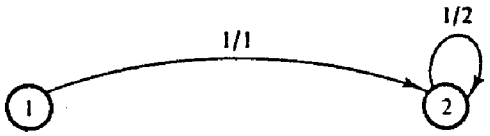

Figure 2.

The third example shows a dynamic system, and the accompanying text makes it clear that the lines denote transfer of something, left undefined, from box to box. There is mention of input values and of output values. The labels on the lines are not properties of the lines; they are for documentation purposes only. (That isn't to say that they are insignificant : it may well make sense to permit documentation to be included in the specification of a line. It does mean, though, that the text does not affect the details of the simulation.)

There is some information in the diagram as to the sort of substance which flows in the lines. Whatever it is can be copied; the value $x_{tt'}$ is sent as input to both the two left-hand boxes. This would not be possible if the line represented a flow of discrete objects, such as people.

The first

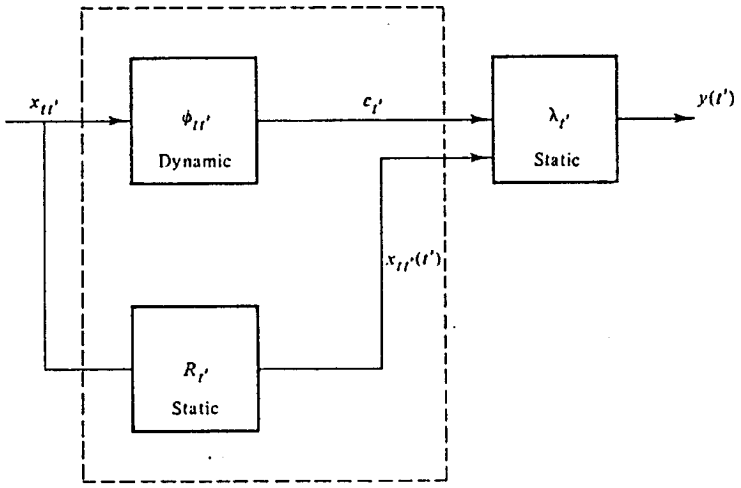

FIG. 2.1

subsystem, denoted by $\phi_{tt'}$ in Fig. 2.1, represents fully the dynamic behavior of the system, while the other two subsystems, $\lambda_{t'}$ and $R_{t'}$, are static and indicate only how the output values are generated from the given states and the current input values.

Figure 3.

Figure 4 (on the next page) introduces some new ideas. It depicts the relationships which exist between a number of propositions P_i. A definition of the function of "an arrow" appears in the text; the intended reference is to the simple solid arrows in the diagram. These lines denote *implication*. The open arrows are not mentioned in the text; they denote *identity*. A third form of relationship between boxes is not denoted by lines at all, but could be : inclusion of one box within another is used to denote *membership of a set*. For the implication and set membership relations, the direction is important; for identity, it is not.

In Figure 5 (also on the next page), there are at least two sorts of line : the horizontal solid arrows, which represent the flow of something copyable, as did the lines of Figure 3. The hollow vertical arrows represent steps in a derivation, and are perhaps to be related to the implication lines of Figure 4. Issues of set membership are also indicated, though, as the whole diagram depicts the transformation of the single box of the top row into the complete diagram of the bottom row. And what are we to make of the lines denoted .... ? They represent *iteration*.

Interrelationships among the properties in Tables 7.1 and 7.2 are summarized in Fig. 4.1. The interpretation of the relationships pictorially represented

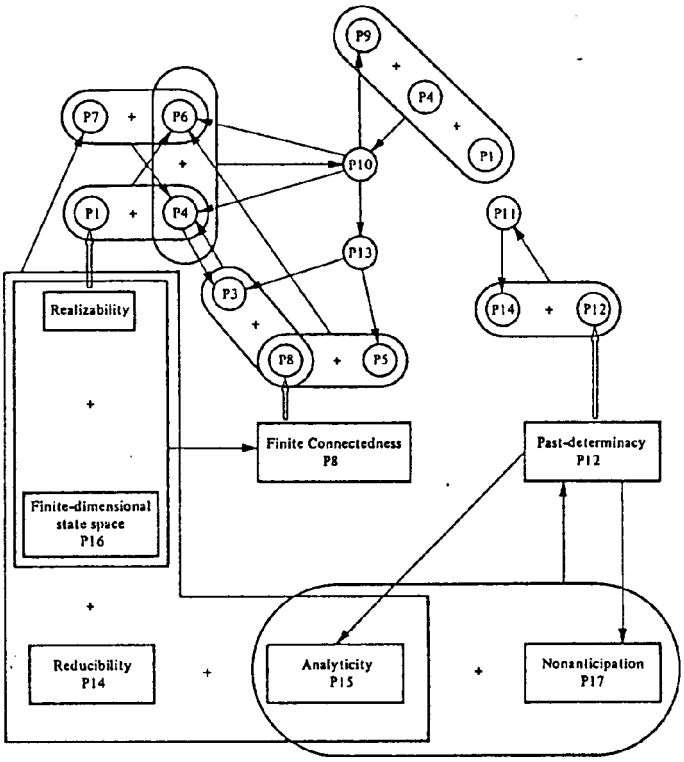

FIG. 4.1

in the diagram is straightforward: An arrow denotes which conditions can be derived by starting from given assumptions. For instance, the diagram shows that if a linear response family $\{\rho_t\}$ satisfies P3 and P8, then P4 is satisfied also; or when P9, P4, and P1 are given, P10 can be derived, etc.

Figure 4.

The results of the principal decomposition theorems in this section are shown diagrammatically in Fig. 6.1.

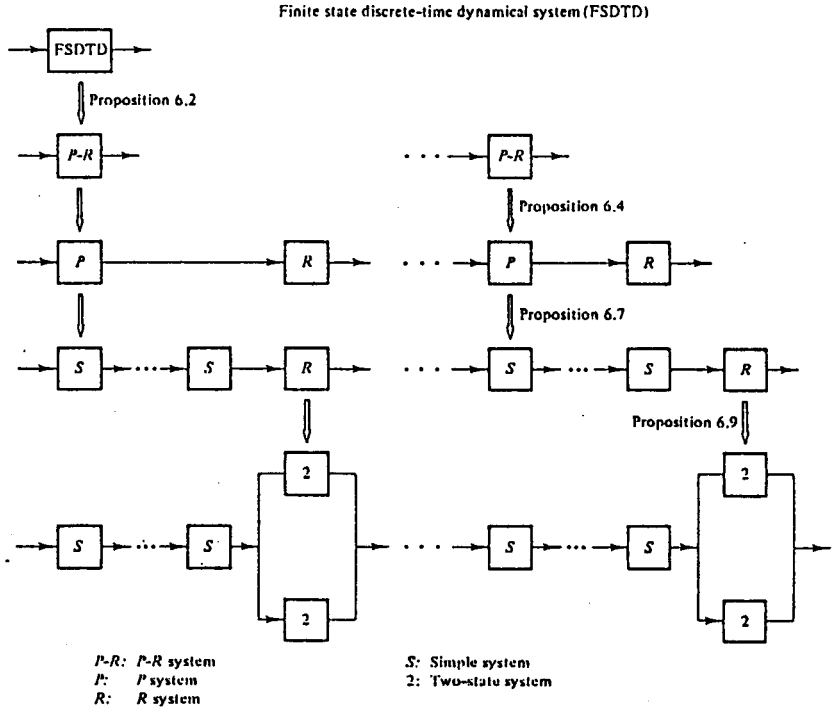

FIG. 6.1

Figure 5.

GENERAL REMARKS ON THE EXAMPLES FROM "GENERAL SYSTEMS THEORY …".

It is interesting that in only one case is there an impression of the strong input-output behaviour assumed in the book itself to be characteristic of general systems - not surprisingly, in the diagram depicting a system studied in the text. (It's true that the selection of diagrams isn't exactly random.) Taking all cases into account, we can list the attributes which lines may or may not have :

Direction : A line may or may not be directed. An undirected line establishes some sort of relationship between the boxes it connects; a directed line also establishes a relationship, but in addition implies that the boxes it joins are in some sense of different sorts with respect to the relationship defined. It may be reasonable to define bidirectional lines which simultaneously carry traffic in both directions, but I'm not too sure about that yet.

Traffic-carrying : A line may or may not carry traffic, which may be material or abstract. A unit of traffic will be called a signal. A line carrying no traffic establishes a correspondence between, or imposes an order upon, the boxes it joins. A line carrying traffic may transfer "things" from one box to another, or it may implement a relationship between some properties of the boxes it joins.

Branched : A line may be simple, joining only two boxes, or branched, when it may link several.

Conservation : If traffic is carried by a line, it may or may not be conserved. If it is conserved, the same number and type of signals must come out as are put in.

Replicability : If traffic is not conserved, it may be replicable. A replica of a signal put into a branched line may be removed from several termini of the line. (I cannot think of any example where more than one copy of the same signal may be removed from a single terminus.)

Activity : The relationship denoted by the line may be continuously active, or may only be significant under certain circumstances.

An attribute which seems to be common to all lines is *type*. Each line means something, or it wouldn't be there; usually, or perhaps always, the meaning only makes sense if the linked objects satisfy certain conditions - so even though "motorcar" and "happiness" seem, inexplicably, to be linked in some people's minds, it is not sensible to connect them with a link meaning "same colour". It is tempting to associate this type with traffic carried by the line; but what sort of traffic implements "same colour" ?

It is interesting to contrast this list with the list in Figure 6 below, taken from the book by Wilson and Wilson[13] only because I couldn't find a more significant list anywhere else. Their classification is designed to apply to information-carrying media, and has essentially nothing in common with my list. Presumably that explains why the characteristics they list are primarily concerned with the practicalities of moving messages about rather than with the abstraction of "pure" lines. I don't know whether or not it's a good argument for the potential relevance of this list that lines, in the sense of this note, are exactly information-carrying media.

(I should add that the book presents a variety of lists, many of which include items which are more or less close to the properties of lines in the abstract. I have chosen to reproduce this one because it is the closest to the sort of list I'd hoped to find; even then, it's clear that the closeness is not very.)

***Table 4-1** Properties of a Zero-entropy Energy Flow*

At any instant:
- Form of energy
- Properties in the space domain—
  - Location
  - Extent in 0, 1, 2, or 3 dimensions
  - Structure—simple or complex wave shape in space
  - Directions of flow of elementary areas of the structure

Any variations of the space structure with time are expected because of some prior knowledge and hence are predictable:
- Properties in the time domain—
  - Location of the changes of the space structure in time
  - Extent (duration) in time of the changes
  - Structure of the changes—a simple or complex wave shape in time for each elementary area of the space structure
  - Power and power level
  - Direction—there is only one in the time domain

Figure 6.

SOME EXAMPLES FROM OTHER CONTEXTS.

SETS. A pair of objects can be denoted by two boxes connected by an undirected traffic-free line. An ordered pair is similar, but with a directed line. A branched line can connect the members of a set of many objects. No traffic flows on the lines, so there is no question of conservation or replication.

ALGEBRAIC QUANTITIES. The lines of an analogue computer circuit are directed and may be branched, with one input and many outputs. (This suggests a "Fanning sense" property of some sort, but I'm not sure how to define it.) The algebraic quantities are replicated as many times as are needed, and each input signal appears at every output. The nodes of a finite element network are joined by lines carrying algebraic quantities, but should probably be regarded as bidirectional : they carry a value from each of the joined points to the other.

WATER PIPES. A network of water pipes is undirected, branched, carries traffic continuously, and is conservative.

STATE DIAGRAMS. The lines of a state diagram are unbranched and directed. It is possible to think of them as carrying a sort of traffic, in the form of a "current state token". (This idea is formalised in the theory of Petri nets.) If so, the lines are only active when the state change they denote occurs.

DATA STRUCTURES. An elaborate graphical notation is presented in the context of database design[14]. Here, lines represent various sorts of logical dependencies of one data item on another, and relationships between them. Some, but not all, are directed; they carry no traffic; some could be considered branched, but the authors introduce additional boxes to avoid branching; and the relationships are continuously active.

SEMANTIC NETS. The arcs of a semantic net are usually unbranched and directed, representing binary relationships between the boxes they link. It is possible in principle to represent ternary or higher-order relationships by using branched arcs, but this seems to be uncommon in practice.

ROAD MAPS. The links are branched, and carry traffic which is conserved. Completely to specify a road system would require directed lines, to allow for one-way streets. Is it sensible to mix directed and bidirectional lines ?

ELECTRICAL CIRCUITS. The lines are branched and carry traffic. Whether or not the traffic is directional and conserved depends on what you want to measure : roughly speaking, current is directional and conserved, but voltage is undirected and replicated. I don't think properties like the failure of replication if the fan-out gets too big is a property which need be included in the properties of lines, as it goes far beyond the limitation of a line to the minimum possible collection of properties.

GENEALOGIES. The lines are directed and branched, with a fan-in of 2 and a fan-out of 1 or more. They carry genes (more generally, inheritance) which are not conserved and may be replicated. If you are following the historical development of the family concerned, they are only spasmodically active; if you are looking at the relationships then they are continuously active.

CONCLUSION, SORT OF.

This description of the lines in a block diagram seems able to cope quite plausibly with quite a wide range of examples. Three questions (or, rather, sets of questions) arise :

1. Is it too complicated ? Can any of the supposed properties of lines be omitted, or perhaps combined, without reducing its descriptive power ?

2. Is it too simple ? Are there any important properties of lines not captured by the description ? If so, what are they, and how can they best be incorporated ?

3. Is it any use ? The original intention was to say something useful for simulation. Have I succeeded ? Why ?

REFERENCES.

1 : M.D. Mesarovic, Y. Takahara : *General systems theory : mathematical foundations* (Academic Press, 1975).

2 : Mesarovic and Takahara[1], page 170.

3 : Mesarovic and Takahara[1], page 11.

4 : T.H. Athey : *Systematic systems approach* (Prentice-Hall, 1982).

5 : Athey[4], page 12.

6 : Athey[4], page 13.

7 : B.P. Zeigler : *Theory of modelling and simulation* (Wiley, 1976).

8 : Zeigler[7], page 10.

9 : Zeigler[7], page 81.

10 : I.G. Wilson and M.E. Wilson : *Information, computers, and system design* (Wiley, 1965).

11 : Wilson and Wilson[10], page 11.

12 : Wilson and Wilson[10], page 29.

13 : Wilson and Wilson[10], pages 48 and 51.

14 : T.J. Teorey, G. Wei, D.L. Bolton, J.A. Koenig : "ER modelling clustering as an aid for user communication and documentation in database design", *Comm. ACM* **32**, 975 (1989).

15 : "Critical research directions in programming languages", *Sigplan Notices* **24#11**, 10 (November 1989).