A METHOD FOR INITIAL FORMULATION OF SYSTEM DYNAMICS MODELS

Dr. R.G. Coyle
Director
System Dynamics Research Group
University of Bradford
England

ABSTRACT

Even the experienced practitioner of system dynamics can encounter serious conceptual problems in getting started on a model, and is tempted to add more and more to his model. A technique - 'list extension' - is described which, from the purpose of the project and the importance of feedback loops, guides the evolution of the simplest adequate model. This model is expressed as an influence, or causal loop, diagram.

The influence diagram should be tested to ensure that its structure contains the necessary elements of a dynamic model. If it fails the test attention is directed to the area of the system where further elucidation is needed.

The techniques have been applied in many practical cases and have been found to give useful results and to increase the efficiency of the modelling process.

TABLE OF CONTENTS

I. Introduction

The first stage in the modelling project is the definition of its purpose. The second stage is the construction of a diagram showing the causal relationships and the model boundary. These diagrams are called 'influence' or 'causal loop' diagrams.

An experienced modeller often seems to find no difficulty in writing down the influence diagram directly from inspection of the system. The novice, or the experienced modeller in an unfamiliar situation, often encounters two difficulties; knowing how to start the influence diagram and knowing how to stop. The result if often a diagram which includes every conceivable variable. This is not good practice; it reflects a poor understanding of how the system operates and an even poorer one of what the study is supposed to be for.

The practice of writing down the influence diagram directly suggests that the model boundary can, in some way, be realised intuitively. It is not, however, clear how this realisation comes about, it is difficult to argue that the model boundary is 'correct' in the sense that it contains enough of the system to generate the dynamic behaviour which has been observed in the system and which is sufficiently important to justify doing the modelling at all, and it is not easy to communicate that boundary-identification skill to the newcomer to dynamic modelling. This paper therefore suggests a procedure which is flexible and easy to apply and which meets the following criteria:-

a) it focusses attention on the purpose of the model
b) starting from a subset of variables which reflect the purpose of the model it leads through successive steps and simple tests to a model containing the minimum number of variables and feedback loops which can be deduced from the stated purpose of the model.
c) it verifies that this minimal model contains only feedback loops, inputs, and outputs, and that there are no loose ends. If the minimal model proves to be inadequate then further detail or sophistication can be added in the appropriate areas so that the model will guide its own evolution.

d) it is easy to learn and practical to apply
e) it is flexible enough to be a servant to, and not a master of, real human modelling skill
f) it leads to tests which verify that no errors have been made in the structural modelling

We now discuss a technique which meets these criteria. This is the LIST EXTENSION METHOD.

II Initial Phases Of List Extension

The list extension method starts from a series of six to eight columns on a piece of paper. From right to left, the columns are labelled the Supplementary List, the Model List, First Extension, Second Extension, and so on.

The Supplementary List contained 'artificial'variables which the modeller has created as indicators, to him, of model performance but which are not part of the system itself.

The Model List contains the names of the variables whose behaviour of the model should explain, or the control of which is aimed at. There should not be more than three to five such variables, and one or two is better at the commencement of a new study.

For each variable in the Model List one writes, in the First Extension column, the names of the variables which most immediately affect it, drawing the influence lines. Variables in the Model List may affect other variables in the same list, as may be the case for any of the lists, and variables in an earlier list may affect those in a later list. The lists must, therefore, be scanned for these connections and the influence lines drawn in. An attempt should be made to add polarity signs to the links.

After the First Extension column has been completed, the 'closure test' is applied to see if a dynamic model has been produced, or if further links are needed to create the requisite feedback loops.

A. The Closure Test

This is a simple procedure for verifying that the influence diagram contains only feedback loops and input structures i.e. that there are no loose ends. If the diagram passes this test then it has the makings of a dynamic model. If not, further detail is not only justifiable, but necessary.

The property of CLOSURE means that a model must contain at least one feedback loop, and that all its variables lie on a loop, have been defined as exogenecus inputs to a loop, or provide supplementary output from a loop.

The test for closure is very simple:

> starting from any point in the influence diagram it must be possible to return to that point by following the influence lines, in the direction of causation, in such a way as not to cross one's track.

This test applies to all points in the diagram, with certain exceptions which are noted below.

The test can be simplified somewhat by exploiting the property itself. This means that, having chosen an arbitrary starting point and traced a path which returns to that point then a number of intermediate points will have been passed and these, of course, lie on the feedback loop just traced out. Since they lie on this feedback loop they lie on a loop and can be dropped from further consideration. One must still apply the closure test to any remaining paths in the diagram to see whether they pass it, whether their variables are covered by one of the exceptions or whether, indeed, the system is not totally closed.

If the influence diagram is not closed, attention moves to the Second Extension list. This contains, for each variable in the First Extension which is not part of a feedback loop, which is not one of the exceptions to the closure rule, or which the modeller can justify representing in more detail, the names of the variables which most immediately affect it. The necessary causal Links are then drawn between the variables in the Second Extension and the variables in the First Extension, together with causal links between variables in the Second Extension list, and between variables already entered in the First Extension and the Model List, and the new variables entering the Second Extension. When the Second Extension has been completed the closure test is again applied, and the process either terminates or continues.

B. An Example of List Extension

The list extension technique really comes into its own in complicated systems. When the system consists of a controller and its complement,* the practical need to satisfy management that a credible model has been built usually forces one to include more detail than is really needed for an adequate model. Even in such cases, it is a good plan to have a discipline for developing the early versions of the model, and the list extension method provides this.

Consider the problem of improving control of profitability and production in a mining enterprise which produces a metal, the market price of which is markedly unstable. Traditional policy has been to produce at a constant rate, but it seems plausible to management that gearing production to price might improve profitability. Management need a design for a production policy which will enable the company to do as well as possible in the face of fluctuating prices. Since the price is so unstable, forecasting seems impossible, so the production is to be tied in to average price. World production is large compared with the mine's output so that this is unlikely to affect the price.

In a mine, preparatory tunnelling, or development, precedes production. Since only limited amounts of production machinery can be deployed in a given developed area the size of the developed reserves affects production. Investment in developed reserves affects profitability, and profitability affects the level of reserves which can be supported financially, thus affecting target reserves. Development produces waste rock which competes for shaft hoisting capacity with the ore from mining.

This is a very complicated problem and one's first reaction might be to produce a very detailed model. Such a model could well be ideal for short-range tactical production planning but that is not what is required. Figure 1 shows how list extension, as it might be applied to this problem, leads to a fairly simple model boundary thus creating the framework for further modelling if the initial model proves to be inadequate. This framework, or model boundary, shows the facets of the system which have to be considered in order to find a closed set of feedback loops. Some of these areas might require modelling in more detail or with greater subtelty but the modeller now has a

---

* Briefly, the 'controller' in a system is the part whose workings can be ascertained exactly, e.g. the firm itself. The 'complement' interacts with the controller but cannot be modelled with certainty, e.g. the firm's market

modelling guide related to management's problem and should be able to produce more useful results than would be the case of the simply built, say, a very detailed production model.

To return to list extension, in this case, the model contains six feedback loops. The first, A, appears at the Second Extension but the diagram is not closed by its emergence because profitability has not been declared to be supplementary variable. A second loop, B, appears at the Third Extension but does not close the model.

The Third, fourth and fifth loops, C and its two unlettered parallel branches, are found at the Fourth Extension. The diagram is, however, unclosed until the detection of loop D at the Fifth and Sixth Extension, at which point it becomes the simplest model which can be built of the system. Whether it is the most adequate model is entirely another matter. It is unlikely that it is, but the modelling process has now started and the model itself, and its output will guide its own elaboration.

The influence diagram, while passing the test of closure, will raise many questions in the analyst's mind about how the system might be changed from its present form to a better one. For example, should the model parameters be fixed or should they be converted into true variables dependent on other parts of the system, perhaps including quantities not yet in the diagram?

From the building of influence diagrams we turn to two procedures which abstract important information from them and check that they are free from certain types of error. These two procedures - Type Assignment and Coherence Testing - will first be described from a theoretical standpoint. Later we consider their use in practical modelling situations.

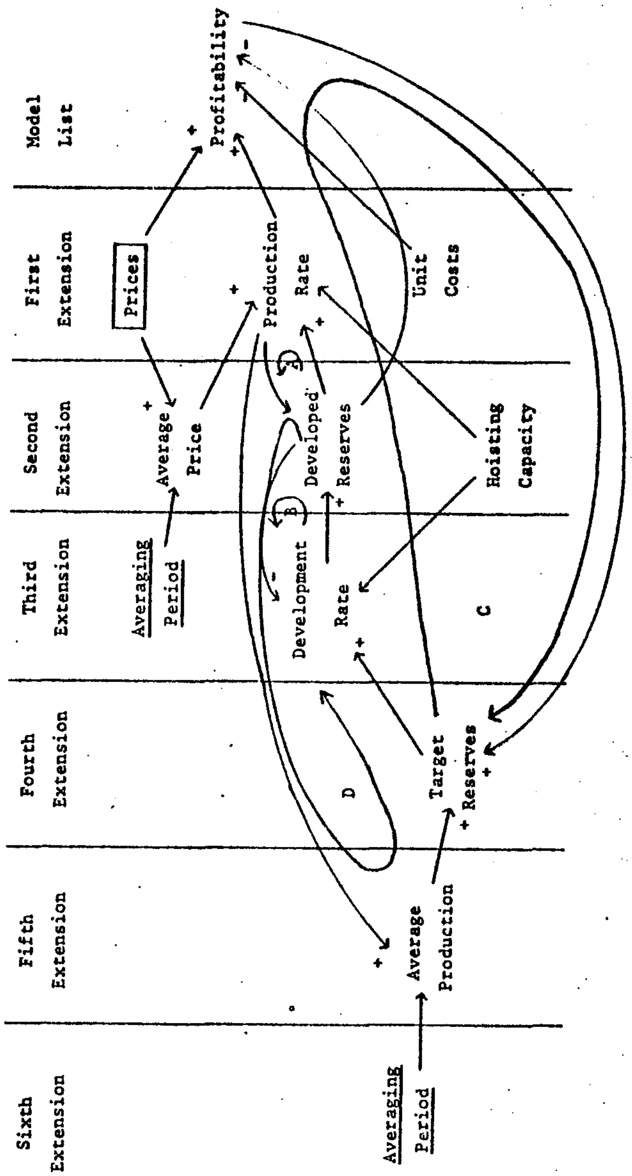


Fig. 1. LIST EXTENSION FOR MINING COMPANY EXAMPLE

Note: i) Underlining indicates a parameter
ii) The box round Prices denotes an exogenous input

III Type Assignment Procedures

It is necessary to decide, for each variable, whether it is to be a level, rate or auxiliary. This step is often taken for granted and it is, indeed, often obvious that a particular variable should be a level. We shall exploit this property in what follows. In practice however, the intervening substructure between levels and rates is often treated in a cavalier fashion and loose modelling can often be covered up by programming dodges. We therefore need a procedure which will determine uniquely the type of each variable and direct our attention to the underlying problem area if a unique type assignment fails to emerge.

We start from the necessary relationships between the variable types, as shown in Table 1.

For most of this section we shall denote variables by letters rather than by names, so as to show the technique of type-assignment unencumbered by preconceptions about a variable's type which might be generated by its name. In practice, a diagram with names should be used.

Table 1. Relationships between Variable Types

The Table shows the relationships which must hold between the types of successive variables in an influence diagram.

Types for Preceding Variable	Variable Type	Types for Succeeding Variables
R	L	A or R
L or A	A	A or R
i) if a delay in the influence link R ii) if no delay in the link L or A	R	i) if a delay in the influence link R ii) if no delay in the link L

Consider a simple model involving only three variables, X, Y, and Z, as in Fig. 2a).

To start the type-assignment process suppose that we have some reason for choosing variable X to be a level, perhaps because it is a stock of completed product. Write the variable type near the name, and enclose it in a box to show that X is the chosen starting point. We can now work either forwards to Y or backwards* because a level can only be preceded by a rate. Thus Z must be a rate. The dotted lines and their directing arrows indicate the sequence of derivation of the variable types and have nothing to do with the direction of causation in the feedback loops.

To type Y we first work forwards from X and then backwards from Z, noting that neither the Y - Z link nor the X - Y link contains a delay, so that the conditions i) for a rate in Table 1 do not apply.

From the X - Y link, Y can be either a rate of an auxiliary, and, from the Z - Y link, Y can be a level or an auxiliary.

We write these conclusions onto the diagram and it is fairly obvious that only if Y is an auxiliary will the forward and backward derivations of its type be consistent.

The type-assignment for this model must, therefore, be

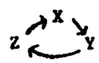
X - Level, Z - Rate, Y - Auxiliary

In this case, the assumption about X led to COHERENT conclusions about Z and Y. The absence of coherence indicates that something is seriously amiss with the model, but its presence only means that the model is 'correct' in a very limited and technical sense.

* Forwards means in the direction of the influence link and vice-versa.

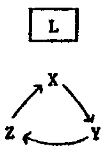
Fig. 2 <u>A Simple Type Assignment</u>

Fig. 2a)



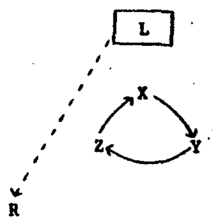
The given influence diagram

Fig. 2b)



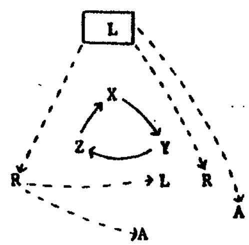
Initial assumption that X is a level

Fig. 2c)



Deduction that Z must be a rate

Fig. 2d)



Determination of the type of Y

In solving assignment problems we have found the following empirical rules to be useful:-

i) if a variable is a level, work backwards first to find the preceding rate,

ii) if a variable is a rate, work forwards first to find the succeeding level,

iii) apply these two steps as often as possible before attempting to assign types to those variables for which there is more than one possibility,

iv) even where a variable's type appears to be unequivocally determined by rule i) or ii), always check if possible by following another path to the variable to make sure that the type assignment is consistent. This will reveal any errors in the influence diagram.

v) In practical modelling, the types of several variables are often obvious and one therefore has several simultaneous starting points. Type assignment then needs to be applied only to the interconnecting structure and its main value is in showing whether or not that part of the model has been worked out in sufficient detail.

vi) Where two variables are connected only by a delay they must be rates which provide further starting points.

The notation includes numbers, written at the side of the dotted lines, to indicate the approximate sequence of the derivation. The numbers are purely explanatory and are not part of the technique of type-assignment. Usually, at any one time, there are alternative steps in the type-assignment process. The numbers do not, therefore, mean that step n <u>must</u> precede step n + 1 but, generally, the lower the number the earlier in the type-assignment process that step should occur.

For a real system, the starting assumption is not made arbitrarily, but on the basis of what is known about the character of the particular variable. A variable which appears, from its name, to have the characteristic of accumulation is probably a level. One which seems to be a flow may well be a rate, but it may be an average or level so in general it is better to start from a variable which is known to be a level, only starting from rates when they are connected by a delay.

As we have defined coherence, it is a structural concept which derives from the pattern of connections in the diagram and, as such, either exists or does not. There is, however, a DEFINITIONAL COHERENCE which arises from the nature of the variables whose types have been inferred from the starting assumption. Thus, in our first example, we assumed that our knowledge of the character of X indicated that it could be a level, and from that we inferred that variable Z had to be a rate.

Now, the statement 'X is a level' derived from our knowledge of X as a system component, and the statement 'Z is a rate', was inferred from the structural relationships in the influence diagram. Clearly, this second statement must also marry with our knowledge of the nature of Z as a system part. If, for instance, Z has all the appearance of an integration then it is very unlikely that it can be a rate, and the influence diagram, although structurally coherent would be DEFINITIONALLY INCOHERENT.

The test for definitional coherence must be applied to those variables which have been inferred to be levels or rates from the starting assumption, once the test of structural coherence has been passed. If any of the inferred levels or rates fails the definitional coherence test then a mistake has been made in drawing up the influence diagram from the verbal description of the system or a definitional mistake was made in the starting assumption. In practice it is almost certain that something has been missed out, probably in the list extension process, and the only solution is to check the diagram

against the system in the hope of discovering the error.

In the remainder of this paper we shall assume definitional coherence.

We shall conclude by working a fairly complex example. The working and the results are given in Fig. 3. with a few explanatory remarks.

Despite the complexity of the influence diagram this is a simple assignment problem, because 6 variables - C, E, J, K, V and T, are fixed by the three delays. There are 6 successor levels to these rates, G, L, N, P, Q and W so that 12 variables are typed automatically. The large number of ticks on the diagram shows the extent to which the system solves itself. The 6 levels immediately give the other rates which precede them, for example, step 17 gives W as a level, which automatically makes Z a rate, step 18. The high degree of self-determination of this system makes it almost too easy to assign the types, and this is not uncommon for apparently large and difficult influence diagrams. This is true for most large models and the value of Type Assignment lies in unravelling the fine structure of a model.

It is important to make sure that all links are checked to ensure system coherence - even if this is done at the end of the assignment calculation as in steps 31 and 32 in this case.

IV. Structural Incoherence in Influence Diagrams

The presence of incoherence means that a mistake has been made, either in the type-assignment process or in the system description as it appears in the influence diagram. We shall devote the remainder of this paper to a treatment of the causes of, and remedies for, incoherence arising from the second of these causes.

Consider the system of Fig. 4a)

Fig. 3. **A Complex Type-Assignment Example**

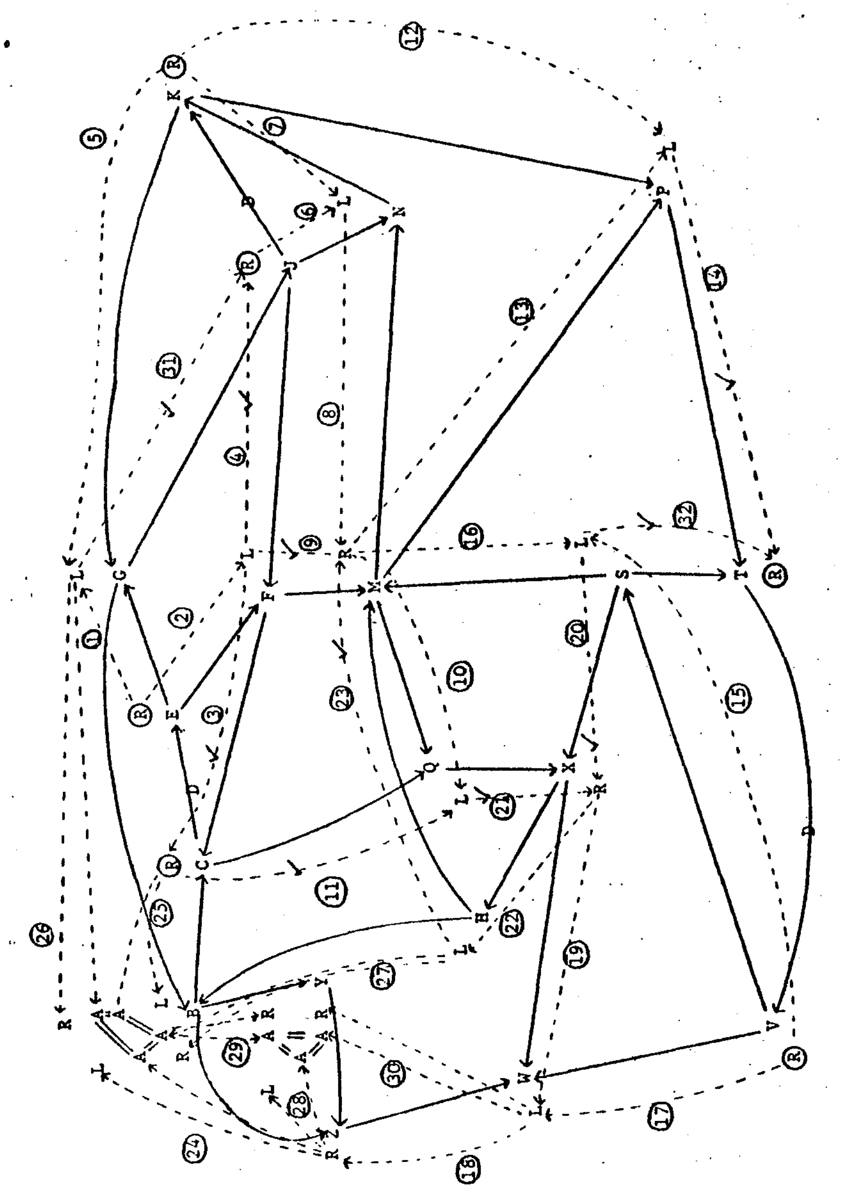
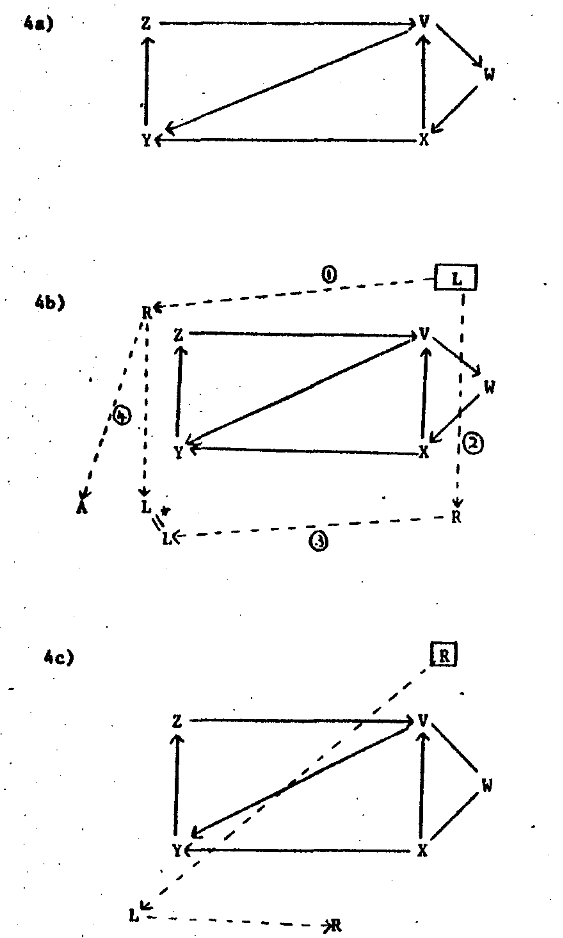


Fig. 4 **An Incoherent Influence Diagram**



With respect to Fig. 4a) this is closed and appears to be an acceptable influence diagram. If V is known to be a level we get 4b).

Step 3 in Fig. 4b) indicates that Y must be a level, and step 4 shows that one of the possibilities for Y is that it may be a level. As far as links Y -Z and X - Y are concerned, Y is a level. However link V - Y shows that Y cannot be a level, and the diagram is incoherent.

The real importance of the concept of incoherence lies in what it tells us about the model as it has been developed. DYSMAP* sometimes allows one to bend the strict rules of system structure far enough to permit the writing of a computable program from an incoherent diagram, so that incoherence does not always prevent apparent progress. It does, however, prevent real progress because the model contains errors which should have been cleared up.

There are two possible causes of Incoherence- either the starting assumption was invalid, or there is some fundamental fault in the modelling. This may be that a link has been put in which does not exist in the real system, or that the influence diagram contains impermissible components.

Taking the first of these possibilities, we examine the other option for V, namely that it could be a rate. Fig. 4c) shows that this assumption also leads to difficulties. Clearly X and V cannot both be rates, as there is no delay recorded for the link V - X and the diagram is still structurally incoherent. The incoherence is more than a matter of the starting assumption and whether there is reason to regard V as a level or, for a different system but the same diagram, to treat it as a rate, the diagram is coherent. In neither case is the reason hard to find, and in both examples, it stems from a misunderstanding of the system structure.

For the first case, where V was assumed to be a level, the solution lies in a further examination of the system. Certainly, the link from V to Y is suspect.

*DYSMAP - Dynamic System Modelling and Analysis Package - is a DYNAMO-type language developed at Bradford.

The second possibility, when V was a rate, is more easily disposed of. Again there has been an error in drawing up the diagram from the investigation of the system. The detected incoherence actually helps by suggesting that there may be a delay in the link from X to V which has been overlooked. If there is, and this can only be determined by further investigation of the system, the diagram immediately becomes coherent.

We have used the same influence diagram to represent two different systems each of which is incoherent because an error has been made in drawing up the diagram from the system description. In both cases the solution of the incoherence is a matter of 'back to the drawing board', that is, further investigation of the system itself is called for. We refer to such cases as SYSTEM-RESOLVABLE.

V. Structurally Resolvable Incoherence

It is sometimes possible to dispose of incoherence by arguments deriving from the fundamental concepts of system structure. This is called STRUCTURALLY-RESOLVABLE INCOHERENCE, and we now consider an example of such a situation shown in Fig. 5.

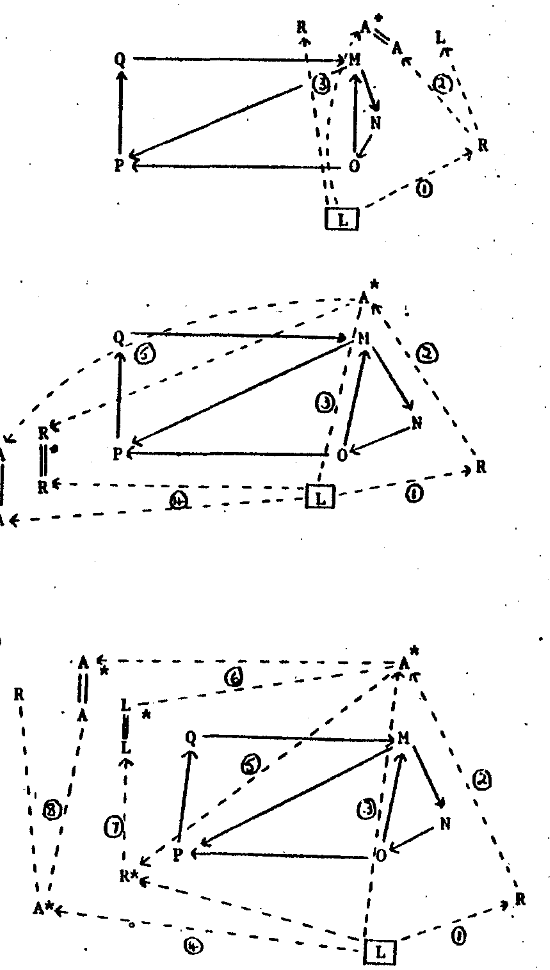
The first stages in the calculation are shown in Fig. 5a), from the starting assumption that O is a level.

In the usual manner we make a provisional assignment of M as an auxiliary and proceed with steps 4 and 5 on links M - P and O - P., as in Fig. 5b).

There are two possible provisional assignments for P; as a rate or as an auxiliary. Definitional considerations will sometimes, but by no means always, distinguish between a level and rate, but not between a rate and an auxiliary, nor between a level and an auxiliary. In order, therefore, to resolve this apparent case of what may best be called semi-coherence we must attempt to type Q and, since there are apparently two options for P, we have two steps for the link P - Q., as in Fig. 5c).

Fig. 5. Structurally-Resovable Incoherence

5a) 5b) 5c)



Unfortunately, both steps 7 and 8 lead to a result which is coherent with step 6 and we can, it seems,make two coherent type-assignments.

Variable	Type	
M	A	
N	R	
O	L	
P	R	A
Q	L	A

This outcome appears to defy the rule that type-assignment should lead to a unique model order. The key lies in the feedback loops as the basic components of system structure.

This system contains three loops, the possible structures of which are:-

Loop	Variable Names	Variable Types		
	M	A		
(1)	N	R		
	O	L		
	M	A		A
	N	R		R
(2)	O	L	or	L
	P	R		A
	Q	L		A
	M	A		A
(3)	P	R	or	A
	Q	L		A

Now a feedback loop must contain at least one rate and one level. Loop 3 will violate this requirement if P and Q are typed as auxiliaries and they must, therefore, be a rate and a level respectively. This reduces the semi-coherence to unique coherence.

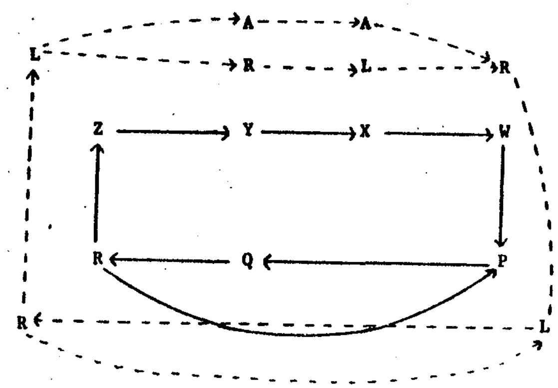
It will be seen from this example that coherence is a property which relates to and derives from the structure of the influence diagram and the fundamental concepts of feedback-loop structure.

VI Double Coherence

It sometimes happens that a system has two or more coherent solutions and there is no way of distinguishing between them by appeals to definitional points or by the use of the structure of feedback loops. For example, consider the system of Figure 6 dropping some of the detailed assignment steps. This is a case in which, between variables Z and W, it is possible to formulate two equally coherent sets of types for the same structure.

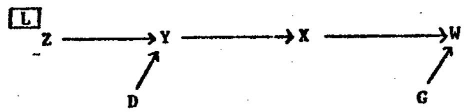
The solution to this kind of dilemma lies in noticing that the parameters of the system have not been included in the diagram. Normally they are not needed because, as we have seen, quite complicated systems can be assigned without reference to the parameters.

Fig. 6. Double Coherence

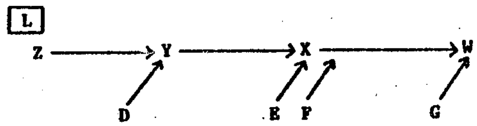


For the disputed top line in this case, consider two possibilities

1.



2.

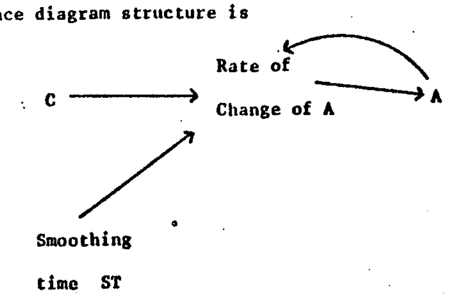


where D, E, F and G may be parameters or input variables (though not the solution interval, DT, because that is a parameter of the simulation, not the system).

In the case 1 the implied equations are:-

$$Y = f(D, Z)$$
$$X = f(Y)$$
$$W = f(G, X)$$

and this means that X must be a level as only a level is parameter-free. A first-order information delay has a parameter, of course, but its correct influence diagram structure is



which is not the same as the above case of double coherence.

A little thought will show that in Case 2 the assignment must be auxiliaries, and that this approach does not depend on the particular number of parameters used in this example. Thus we may restate the position that the type assignment must be unique providing enough detail is included in the influence diagram.

VII Application of List Extension, Type Assignment and Coherence Testing in Practical Modelling

In many cases of practical modelling the influence diagram becomes very large. There are two principal reasons for this:-

a) Many practical problems really are very complicated with large numbers of interacting variables, and it is not easy to see which variables are essential and which are, relatively unimportant.

b) In order to have a chance of recommendations being implemented, and that is always the real object of the exercise in actual business situations, it is essential for the managers concerned to have a high degree of confidence that the relevant factors have been modelled.

The advantage of the influence diagram is that it makes the model structure very clear and this is an aid to elucidating the system structure from management, who are in the best position to know what that structure is. However, the diagram also shows what has not been included, and this gives managers the opportunity of being the arbiters of relevance.

Simulation modelling makes it easier to include a factor to satisfy a manager who feels that it is relevant to system performance, than to convince him that it is not and may be omitted from the model. In any case, the analyst will usually find it hard to adduce convincing arguments for the irrelevance of a factor until it has been included and tested.

In practice then, we are likely to have large influence diagrams, and this has three drawbacks:-

a) The type-assignment and coherence-testing procedures become rather tedious yet if not applied there are more chances for errors to creep in.

b) The larger the model is, the more difficult, time-consuming and expensive it will be to analyse and understand.

c) As the model size increases, the problem of conveying it and its results to management in a concise, comprehensible, fashion in the short time which is usually available for presentation of results becomes almost involvable. The virtue of simulation - getting management confidence in the model - almost proves to be its downfall when to the influence diagram is added a welter of computer printout and system analysis.

These methods have been applied in a number of very different practical projects in tanker chartering, metals manufacture, chemicals, and consumer goods industries and have been found to be well worth the trouble involved. Further details appear in Coyle (1976) and more examples in Coyle & Sharp, (1976).

References

Coyle, R.G.	1976	Management System Dynamics University of Bradford and John Wiley
Coyle, R.G. & Sharp, J.A.	1976	System Dynamics Problems & Cases University of Bradford

Prepared: 15th April, 1976