BAMBOO

A BEHAVIOR ANALYSIS EXPERT SYSTEM FOR SYSTEM DYNAMICS MODELS

Andreas Kleinhans

Betriebswirtschaftliches Institut
Universität Stuttgart
West Germany

Abstract

There are many ways to combine Expert Systems and System Dynamics. In a short overview the paper will show useful basic combinations. As an experimental project BAMBOO will be introduced. It is primarily designed to test the usefulness of descriptive knowledge processing techniques for building and using System Dynamics Models. BAMBOO holds expertise of all SD-objects and structures, their possible combinations and the behavior they cause. BAMBOO generates the necessary knowledge about the user model by a system driven dialog. On the basis of this knowledge it shows the conclusions the model implies. For instance, BAMBOO determines which variables are sensitive and how the model will propably respond during the simulation.

System Dynamics and Expert Systems

Expert Systems are the most promising tools which have evolved in the field of Artificial Intelligence in the last few years. We use the term "Expert System" in its original sense, i.e. for an intelligent system which relies on descriptive knowledge (organized for instance in "facts" and "rules") and explains its reasoning. Some papers have recently discussed the combination of Expert Systems and simulation such as Shannon et al. (1985), Gould (1985), Uschold et al. (1984), Futo et al. (1983) and O'Kneefe (1985). Examples for implementations are SMARTD (Gottinger (1985)) and KBSIM (Young (1985)). A very convenient and user-friendly system is STELLA (Richmond (1985)), it is however not an Expert System. The term "Expert System" is not yet used uniformly.

In the first part the paper will introduce a classification of SD modeling processes and possible tools an intelligent system may offer. We want to distinguish between the system support as such and the way it is implemented. This means that every tool can be implemented as an expert system or a plain assembler program.

There are many steps in a System Dynamics modeling process and even more terms to describe them. They are usually classified according to the sequential phases. For our purpose an organisation according to the three main modeling problems,

construction, aquaintance and consultation seems to be more appropriate (cf. Fig.1). They overlap one another. The construction process includes model design and verification. This step requires qualitative and quantitative adaptation, i.e. parameter changes and changes in the model structure. A user becomes acquainted with his model or with a prebuild model by primarily making quantitative changes. The consultation process is goal oriented and answers the question: by which actions can I reach the goal x? These actions will mainly result in qualitative changes.

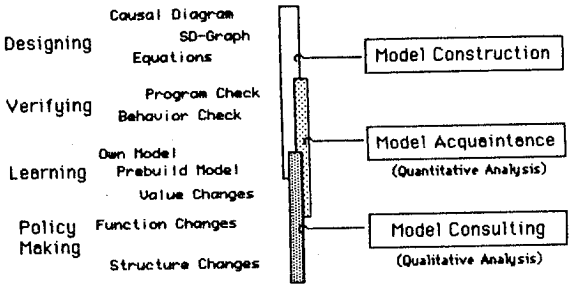

Fig.1 Modeling Processes

There are four type of tools which support these three processes (cf. Fig.2). They can be combined in any way. The first type supports the man-machine-interface. It offers a user-friendly dialog environment for both the technical input-output-process (e.g. windows and mice) and a user-adapted modeling starting point. STELLA for instance heavily relies on this feature. A sophisticated version would even be able to support a user formulating his mental concept.

Since the model should be based on real data it would be convenient to have an automatically adapting tool, which could be called a validation optimizer. The output would be an "optimized" model, i.e. a model which reproduces the real data as accurately as possible. The validation optimizer requires a strong feedback between the design and validation processes. Research in this field has been done by Keloharju (1983), Barlas (1985), Coyle (1985) and others.

The knowledge-extractor is a highly multi-functional tool. Although the essential model-knowledge is accumulated in the model-equations, it is not directly accessible to the system, because of its coded form. When the system wants to use this knowledge it has to make a description of the model, i.e. it has to be able to treat the model as data. In combination with other tools it can form a sophisticated knowledge-based system. The typical implementation of the knowledge-extractor is a rule-based expert system.

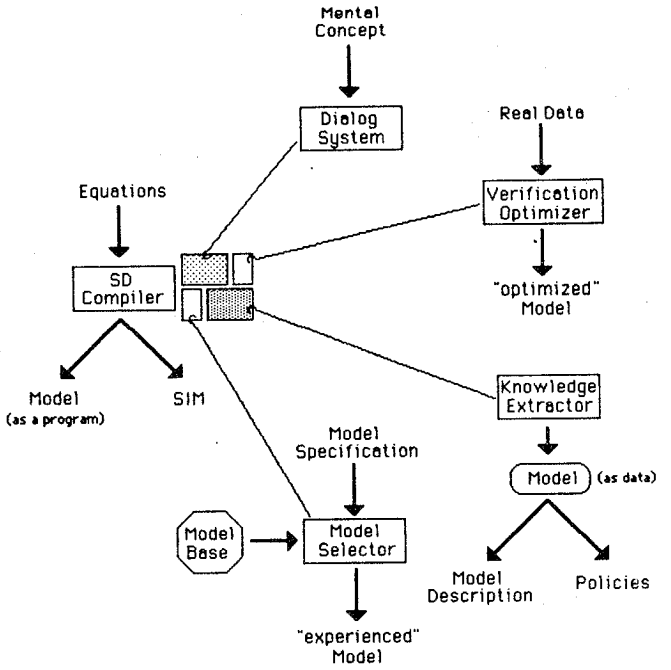

Fig.2 Modeling Tools

The knowledge-extractor mainly incorperates SD-knowledge. As a result the user does not have to know SD very well. He might not even notice that he is working with a SD-model.

Another type of tools focuses on the incorperation of application-knowledge. These tools can be considered as model-selectors which rely on prebuilt modular model-bases. Since they already "know" several models the user just has to tell the system the details of his problem. The output is what we call an "experienced" model, because the model is composed of modular pieces which are already tested and well known.

We can combine these tools in many ways according to our system philosophy. All combinations can be regarded as an evolutionary set of systems. On the lowest level we can find the "plain" SD-compiler which gives almost no support, e.g. Micro-Dynamo. An example of a sophisticated system on the highest level could be a knowledge-based application expert system. It would offer an ultimate user-friendly model-construction support with access to a large model-base. It would make behavior analyses, give policy advises, and offer learning tools for beginners etc.

The BAMBOO project

BAMBOO is designed for an advanced system. It combines two tools, a dialog system and a knowledge-extractor (cf. Fig.3).

BAMBOO focuses on the main purposes of a simultion system: the prediction of the future behavior of the real system and the derivation of policies for intended goals. BAMBOO analyses the behavior of the model and offers this knowledge to the user. In addition to this the user is able to speed up the verification and learning process.

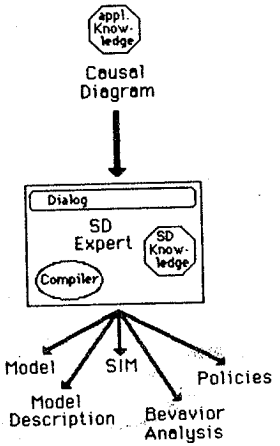

Fig.3 BAMBOO System

The main part of BAMBOO is a descriptive knowledge-base organized in facts and rules. It holds SD-knowledge (rules) and a descriptive representation of the user model (predicative facts).

Examples for SD-Rules are:

```
Rule S12:  If x has_an_inputrate or
              x has_an_outputrate
           then x is_a_level

Rule S39:  if y is_a_loop
           then one_element_of y is_a_level.
```

Examples for model-facts are:

```
STOCK is_a_level
STOCK is_a_level>0
S_IN is_a_rate_for STOCK
ORDER +implies S_OUT
```

There are basicly two ways for the system to generate the necessary basic facts of the user model. The system can either ask the user by a system-driven dialog or start an equation-parser if the equations are already in the system. We noticed that many SD users do have problems when they put their mental model concept in System Dynamics code. This is why we designed BAMBOO to focus on the modeling support as early as possible. The support begins at the causal diagram. The system guides the user to transform the diagram into DYNAMO-like equations. BAMBOO can generate its knowledge-base simultaneously.

Fig.4 shows the BAMBOO system and its environment: the simulation, the real and the mental system. The dotted ellipses represent essential knowledge-pieces. The counterpart of the equations are the model-facts of the BAMBOO system.

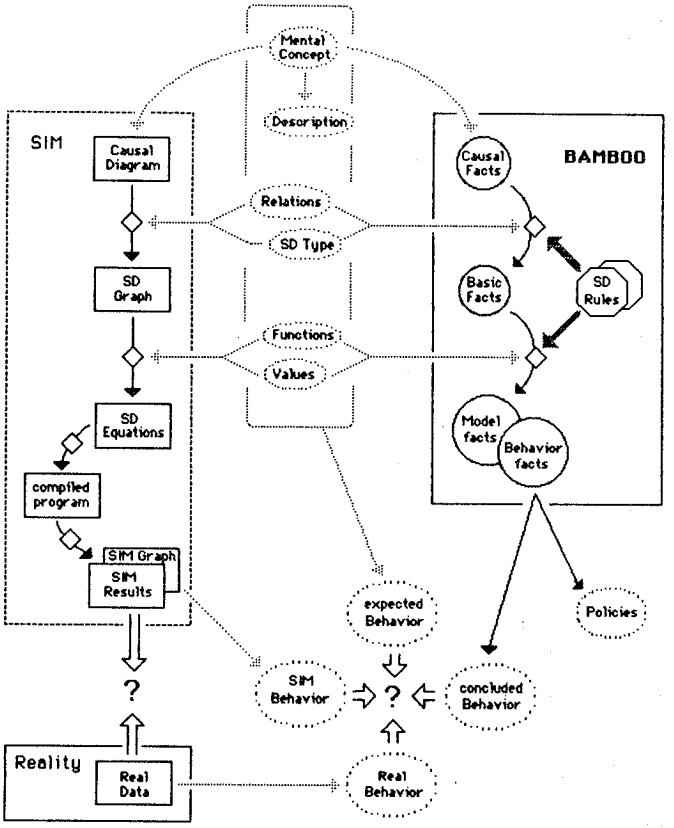

Fig.4 BAMBOO and its environment

These four systems produce statements on four possibly different types of behavior:

1. the real behavior (which is however past-oriented only)
2. the expected behavior (which is more or less based on intuition)
3. the simulated behavior (by SD)
4. the concluded behavior (by BAMBOO)

The behavior of the real system is of course the most important one. The other statements have to be compared with it. The "expected behavior" is probably always the one which differs most (otherwise we would not have to simulate). The simulated behavior should come very close to the real one (otherwise our model would be wrong). Since BAMBOO can be regarded as "simultation" of the model-simulation, the concluded behavior should describe the SD-simulation output as accuratly as possible. (Indeed, there would be no need for a SD-simulation, if BAMBOO directly simulated the behavior of the real system.)

Descriptive Behavior Analysis

Descriptive programming has many advantages. It is however not clear whether this is a useful technique for behavior analysis or simulation-modeling at all. Therefore BAMBOO is in the first place an experimental project which is to find out how the user can be supported by a descriptive and declarative knowledge-base.

This concerns the general problem of any simulation: how exact has the description to be in order to obtain an adequate representation of the real system and to be able to make useful predictions. A model is usually completely represented by its equations. When we choose descriptive programming we have to consider an additional level. All together there are three problem solving situations:

1. the predicative description of the equations is working
2. the evaluation of the equations is necessary
3. the System Dynamics model itself is too vage

In other words, there are models which can be simulated by a descriptive technique just as good as by SD. There are other models whose equations can not adequately be described by predicative facts. And there are models whose representations in SD itself are not appropriate. These models can usually proof anything.

BAMBOO tries to solve this problem by a subsystem with the following three parts (cf. Fig.5):

a. sensitivity detector
b. vageness detector
c. conflict resolution component

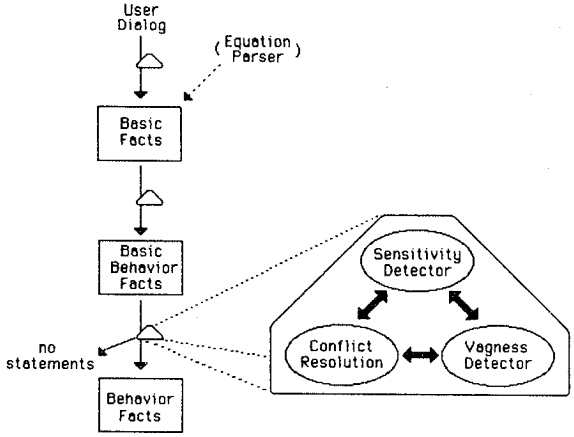

Fig.5 Behavior Analysis Subsystem

The basic model facts are available at the end of the initial dialog. Then BAMBOO calculates the so-called basic behavior facts. They mainly describe the behavior of those variables which are not in conflict with one another.

Examples for some rules are:

```
Rule B2:  If x is_a_constant
          then x is_stabil

Rule B9:  If x is_a_loop and
             y is_an_element_of x and
             y is_converting
          then x is_converting
```

The sensitivity detector is activated when certain variables or loops show contrary behavior. It tries to determine the conflict domain. In case of success it passes control to conflict resolution. It is useful to combine the sensitivity detector with a quantitative method. Forrester (1983), for instance, presented a method by which significant variables and feedback loops can be identified.

In case of failure the vageness detector is activated to find out whether a bahavior analysis still makes sense, whether a simulation is inevitable or whether the model is ill designed. The bevavior analysis fails, if the model is recognized as too vage. Otherwise conflict resolution is reactivated in order to find the best solution possible. In this case the system can still conclude some useful statements. Although they can not be used for predictions, they indicate the problems the user should pay attention to.

Some examples for possible concluded facts are:

```
$_RATE is_oszillating
$_RATE is_sinking_initially
$_RATE is_sensitive_for ORDER in_domain 42,346
ORDER is_growing_initially
ORDER is_not_predictible
```

The user can get access to the concluded facts by asking the appropriate questions. He can also ask for some policy advices.

The following example shows a possible session. The user wants the ORDER curve closer to the SUPPLY one. The user questions begin with "?":

```
? ORDER may_be_closer_to SUPPLY

Did you try_to_change $_RATE in_domain 42,346?

? why

I try to solve:
RULE P5:   if X +implies Y and
              Z is_sensitive_for X in_domain D and
              try_to_change Z in_domain D
           then X may_be_closer_to Y

because I know:
           ORDER +implies SUPPLY and
           $_RATE is_sensitive_for ORDER in_domain 42,346

I ask you:
           try_to_change $_RATE in_domain 42,346
```

Although the policy-generator in this example is implemented in a very simple way, it is quite useful. However not every policy can be coded so easily.

Conclusions

Since System Dynamics is mainly interested in statements about the behavioral responses of the models, it seems reasonable to support the user by a descriptive and declarative knowledge-base. Descriptive programming techniques are the basis of common expert system tools. The main purpose of the BAMBOO project is to find out the possibilities and limits of supporting System Dynamics Modeling by these expert systems. Although not every model is completely describable the user can reach a faster and deeper understanding of his model. More research is required in order to improve the rules and the dialog-interface.

References

Barlas, Y. (1985). Comparing the Observed and Model-generated Behavior Patterns to Validate System Dynamics Models. Proceedings of the 1985 International Conference of the System Dynamics Society, Keystone, Colorado, pp. 32-47.

Coyle, R.G. (1985). The use of optimization methods for policy design in a system dynamics model. System Dynamics Review, no. 1, pp. 81-91.

Forrester, N. B. (1983). Eigenvalue Analysis of Dominant Feedback Loops. Proceedings of the 1983 International System Dynamics Conference, Chestnut Hill, Massachusetts.

Futo, I., and Gergely T. (1983). A logical approach to simulation (TS-Prolog). In: Wedde, H. (Ed.). Adequate modeling of systems. Springer, Berlin, pp. 25-48.

Gottinger, H. W. (1985). SMARTD: An Intelligent Decision Support Tool for Strategic Management on Technology Diffusion. Strategische Planung, vol. 1, pp. 261-275.

Gould, J. M. (1985). Artificial Intelligence: A Tool for System Dynamics. Proceedings of the 1985 International Conference of the System Dynamics Society, Keystone, Colorado, pp. 317-330.

O'Keefe, R. M. (1985). Simulation and Expert Systems - A taxonomy and some examples. Q.S.S. Discussion Paper No. 70. University of Kent at Canterbury.

Keloharju, R. (1983). Relativity Dynamics. Helsinki.

Richmond, H. (1985). STELLA: Software For Bringing System Dynamics To The Other 98%. Proceedings of the 1985 International Conference of the System Dynamics Society, Keystone, Colorado, pp. 706-718.

Shannon, R. E., R. Mayer, and H. H. Adelsberger (1985). Expert systems ans simulation. Simulation, June 85, pp. 275-284.

Uschold, M., N. Harding, R. Muetzelfeld, and A. Bundy (1984). An intelligent front end for ecological modelling. In: Shea, O. ECCAI 1984. Elsevier, Chichester, pp. 761-770.

Young, D. F. (1985). KBSIM: A knowledge based bool and its use in model preprocessing. Proceedings of the 1985 International Conference of the System Dynamics Society, Keystone, Colorado, pp. 1070-1080.