HBM-view: Designing a Model Viewing Tool for Behavioral Scientists

Tylar Murray
University of South Florida
Department of Electrical Engineering
tylarmurray@mail.usf.edu

Daniel Rivera
Arizona State University
Control Systems Engineering Laboratory

Eric Hekler
Arizona State University
School of Nutrition and Health Promotion

D. Spruijt-Metz
University of Southern California
Center for Economic and Social Research
Los Angeles, California

Andrew Raij
University of Central Florida
Institute for Simulation and Training

***Abstract*—In this paper we present relevant definitions and design considerations relevant to human-behavior modeling software. The guidelines presented here are based off of a user-survey and expert-panel review performed in development of the BehaviorSim model-building tool designed especially for use in behavioral science. Lessons learned through iterations of this tool and unique considerations for those targeting behavioral scientists are highlighted. Our initial survey of 12 behavioral scientists reveals the diversity of opinions and approaches to behavior modeling within the community and gives context to this emerging area of research. In addition to these guidelines, a theory-agnostic method for defining Human Behavior Models (HBMs) is proposed and techniques for supporting different modeling paradigms within a single user interface are discussed.**

I. Introduction

A. Problem Statement

The increasing prevalence of wearable technologies and user-behavior collection, behavioral researchers are becoming increasingly overwhelmed with data.

State-of-the-art user interface often predicts user behavior in order to preload data or to customize the application. Researchers have seen the potential for optimizing a behavioral intervention to suit not only the user, but also the context. Just-in-Time Adaptive Interventions (JiTAI) promise to provide the optimal nudge at the optimal time to aid users looking to change their behavior Imagine, for example, an anti-stress app which knows not to interrupt your work meetings, but also knows when to play your favorite song to help you relax on the drive home. Or consider a smoking cessation app that knows precisely when and where you are most likely to crave, and distracts you with a game of sudoku before you even realize you were about to want a cigarette. The impending wave of context-aware, affective computing applications seems to be the holy grail of behavior change, but researchers are increasingly finding that our models of human behavior are not ready. The search space of designing an intervention for even a single user can be prohibitively large,

Though the machine learning approaches which are enabling detailed behavior-prediction can be applied to behavioral research problems, the inner workings of these models often do not tell researchers much about how the human system works.

On the other hand, psychological models of human behavior act as a guide for behavioral scientists looking to predict behavior, but are rarely computational in nature - making their application in adaptive systems impossible.

A hybridization of these two approaches may allow researchers to leverage the strength of each as they are applied in adaptive behavioral interventions.

B. Human Behavior Modeling

A Human Behavior Model (HBM) is any computational model which attempts to define the human systems conversion of measurable contextual information into measurable behavior outcome. HBMs are a set which intersects the set of computational cognitive models, both sets include approximations of cognitive processes, but in contrast with cognitive models, HBMs put a greater emphasis on the measurable in and outflows of the human system. Cognitive models traditionally have focused on creating a comprehensible reprentation of cognitive processes, but HBMs have no such restriction and require only that a model define behavioral output as some function of contextual input. Thus HBMs include formulations which may not be considered a cognitive model. For instance, a model which offers little more than statistical relationship between contextual input and behavioral output may be considered an HBM.

For our purposes, we classify HBMs uncovered by our literature review into three classes based on intended application 1) Automation Agents - models developed to act as automated decision agents, 2) Emergent Behavior Studies - models developed to identify emergent behavior of a population, and 3) Cognitve HBMs - models developed for prediction and understanding of an individual's behavior. Note that these categories are not mutually exclusive - a single HBM could theoretically fufill all three roles, but the requirements for each category are quite different, and thus existing models in each category are usually quite different.

* TODO: how do HBMs address the issues raised in above problem statement? * why are they important for the future behavioral research? [**?**] * how do they fall short?

C. Existing Tools

Our review of existing tools and publications in the area of behavioral modeling has uncovered three types of tools: 1) Dynamical System Modeling Toolkits, 2) Cognitive Architectures, and 3) Agent Modeling Toolkits. Though each approach seeks to address the problem of human behavioral modeling, the intended use of the model is very different. Thus each approach makes an approximation of the human system from a different perspective, and the resulting models show little resemblence. These three types of tools are in partial alignment with the aforementioned three classes of HBMs.

1) Dynamical System Modeling: Works in this group focus on generalized model-building. Applications include network and business analysis as well as semi-physical system modeling. One might be able to use these systems for modeling a human agent, but a more rigid model architecture for organizing the flow of information is needed. One of the contributions of the behaviorSim toolkit is the implementation of a general structure of information flow through a human agent.

Specialized modeling toolkits are often developed to create a variety of more detailed models focused around a single modeling goal. This is precisely the goal of the behaviorSim toolkit. In this search specialized toolkits focusing on neurological simulation, molecular dynamics, product inventory flow, economics, astrophysics, neurological networks, and power system modeling were all encountered in this search. Works on more relevant topics such as social interaction and behavioral modeling are highlighted in the spreadsheet and warrant further investigation.

2) Cognitive Architectures: The next group is a large set of modeling toolkits which aim to create artificial intelligence through the modeling of cognitive processes. Though a well implemented artificial intelligence will serve as an excellent model of human behavior, such an implementation is far from feasible at this time. The behaviorSim toolkit takes a more empirical approach to modeling cognitive functions than existing cognitive architectures.

3) Agent Modeling: Agent modeling has proven useful in many industrial and commercial applications across multiple disciplines. Agent models are generally developed for one of two purposes: 1) to act as an intelligent actor in an automated task, or 2) to observe the emergent characteristics of many interacting agents. The behaviorSim toolkit does not aim to accomplish either of these goals. It is for this reason that the common BDI agent architecture is not suitable for modeling of human behavior based on empirical evidence.

II. Case Studies

Here we present findings from multiple iterations of development of a tool designed to help behavioral scientists forumlate computational models of human behavior. Evaluation at each step of this iterative process involved a small number of testers as guided by http://www.nngroup.

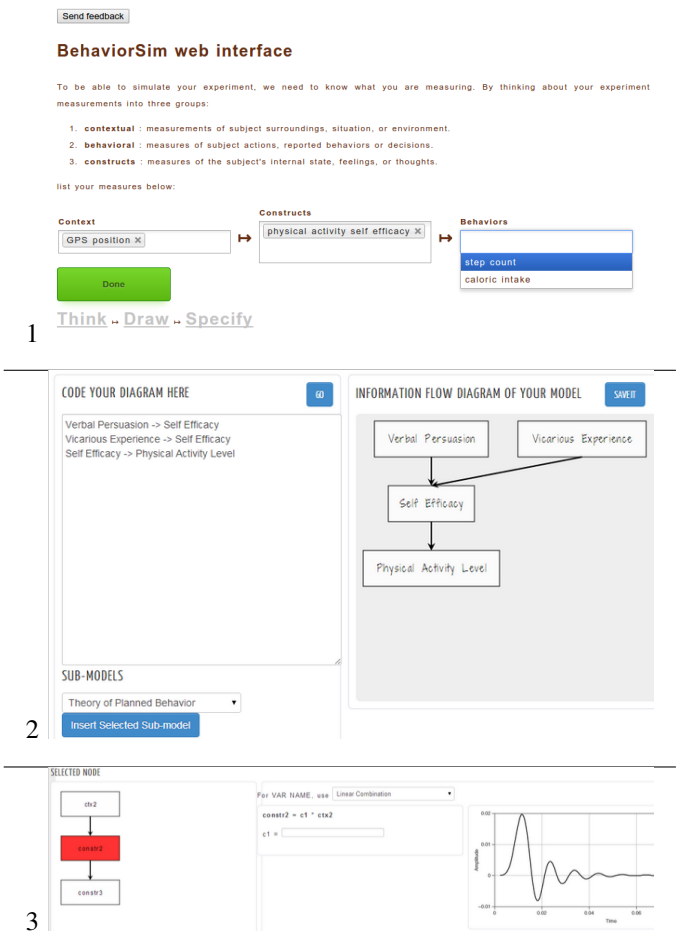

Fig. 1. behaviorSim Model Builder v0.1.x separated model-building into three separate stages: 1) think, 2) draw, 3) specify.

com/articles/why-you-only-need-to-test-with-5-users/
).\subsection{Paper-prototypeModel-BuildingTask}Apaperprototypeserv
andwasevaluatedinformallythroughdiscussion.Thisprototypeallowedustoic
wefoundthatcarefulchoiceofterminologywasimportant,
sincesomeoverlappingtermswerethecauseofsignificantconfusion.Usersalso
anduserseffortswerefocusedondefiningtheinnerworkingsofthehumansyster
sAnnualConferece.Outofthese,
approximately12surveyswerereturned.Ingeneral,
usersstillhadtroublewithmodelingexercise.Mostdidnotstrayfarfromthegive
andotherspravidedverydifferentsolutionswhichdonotfitthe‘‘engineer’
sview’’oftheproblem.Inthesurveyquestionsparticipantsreportedthatthemat
andnearlyallparticipantsexpressedadesireforincreasedcollaborationbetwee
whataretheoutflowsofeach?specify-Howexactlycaneachconstructbecalcula
delays,etcuntiltheconstructbehavesasyouthinkitshould.Dolotsofthoughtexp
whatifs’ofyourmodel.Usesystemidentificationtechniquesondatatoinformy
p-values’.\begin{figure}[!t]\centering\
includegraphics[width=0.9\columnwidth]{img/behaviorSim_
process}\caption{Thefivestagesimplementedinv1ofthebehaviorSimModel
}\label{model-builder-tutorial}\end{figure}Thisaplicationwastestedusin
draw, specify, etc...), the step-wise design is difficult for users who are seeing the process one-step-at-a-time. TODO: review my notes for more

The first iteration of the behaviorSim Model-Builder (see figure 1) took a step-wise approach to the task, which made backtracking difficult.

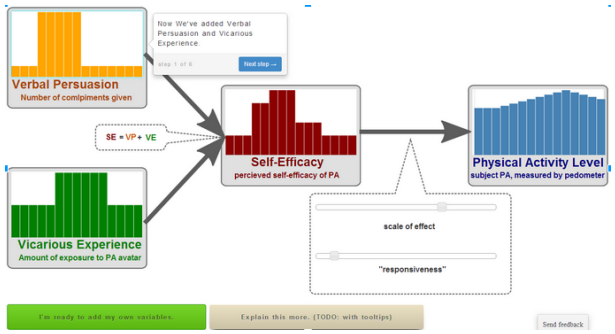

Fig. 2. behaviorSim Model Builder tutorial attempted to merge time-series and information-flow graph representations.

A think-aloud assessment of this first version performed by four domain experts showed that in this design users have difficulty understanding how earlier choices related to later results, and feel constrained by previous choices rather than backtracking to revise the model.

A. *behaviorSim Model-Building Tutorial v1*

Tutorial designed to help users see the bigger picture before diving into the step-wise process. Think-aloud protocol with 2 behavioral scientists and 1 HCI expert.

Lessons learned: TODO: See notes for more

B. *behaviorSim Model-Building Tool v2*

Think-aloud protocol with 2 behavioral scientists and 1 HCI expert, 1 behavioral modeler

Lessons Learned: TODO: see notes for more

III. Design Guidelines

Using lessons learned from our attempts at developing a model-building tool for behavioral modeling we propose a graph-based HBM specification, and provide generalized guidelines for developing modeling and simulation tools for behavioral scientists.

A. *HBM info-flow graph*

A ”standard” approach to describing, designing, and visualizing human behavior models for personal forecasting is needed. There are countless approaches to systems modeling, but concepts and methods ought to be incorporated and fused when possible in their application to behavioral science. The resulting specification will allow for the formal description of a Human Behavior Model (HBM).

We will begin with the most basic definitions and then progress into the intricacies and usage examples of the proposed model-modeling paradigm.

A model is comprised of a network of variables and their interrelations. In order to visually represent the connectivity of the model a network graph can be used. Network graphs have many applications across multiple disciplines, and though the different implementations can be visually similar they can differ significantly in their meaning. In general, however, a graph is comprised of nodes and edges which represent the variables and their connections, respectfully.

We argue that the most intuitive representation is one which uses a directed graph wherein edge arrows to represent the flow of information between nodes. Thus, a HBM directed graph edge from node A to node B indicates that information flows from node A into node B.

$$A => B$$

Graph 1: Can be read as ”A influences B”, ”A informs B”, or similar

This choice of notation is in agreement with graphs used in information theory, communications models, and behavioral science. In contrast, some graphing paradigms (such as probabilistic graphical models) prefer to use notation wherein an edge is used to represent dependency. In these paradigms edges may be read from tail to head as ”depends on” or similar.

While the network graph does an excellent job showing the connectivity of a model, it fails to indicate the meaning of each connection. In the majority of existing applications, the mathematical form of the relationship is implied or else it is neglected completely. For instance, path diagrams from the behavioral sciences frequently denote dependence and do not specify functional form. Adding even further to the confusion is the notion that these graphs are often developed using different statistical analyses which may make different assumptions about the functional definition of inter-variate dependency. The most common analyses assess linear relationships between variables, and thus it is perhaps reasonable to assume that this is the intention of most authors. Assuming this is the case we can return to our simplistic example in Graph 1 and interpret the implied relationship as:

$$B = coeff_ab * A + const$$

Equation 1: coeff represents the correlation coefficient which relates A to B, and const represents a scalar constant

For nodes with multiple inflow edges, such as node B in the following graph:

$$A => B <= C => D$$

Graph 2: Example; now with more nodes!

The resulting formulation is simply a sum of the inflows:

$$B = coeff_ab * A + coeff_cb * C + const$$

Equation 2: Combining multiple inflows via sum is referred to as the superposition principle and is the defining characteristic of linear systems (not to be confused with this linear formulation)

Using this formulation, the general form of our HBM is expressed via the network graph alone (perhaps along with a statement about what edges mean). To express an ideographic implementation of this general model, we must also include a table of coefficient values. This representation is useful, but also very limited. One important feature which this formulation does not take into account is the dynamics of the relationship. This is very important for human behavior modeling because the variables in a HBM will often change their value over time. Taking time into account our formulation for Graph 1 becomes:

$$B(t) = coeff_ab * A(t) + const$$

Looking at this closely we can note that at each point in time the value of A depends only on the value of B at that same instant in time. This assumption is fine for many applications, but we argue that this is a very poor assumption for human behavior models. What if A is influenced by B but there is some lag before the effect manifests? What if A is influenced by B, but only if B is above some threshold value? What if A is influenced by the rate of change in B rather than the value of B itself? All of these scenarios cannot be expressed using this simple linear relationship.

To resolve some of these issues, a differential equations can be used to describe the relationship between variables as described by (CSEL paper which connects path diagrams and fluid-flow). Using the differential formulation our equation for Graph 1 becomes:

$$B(t) = TODO..$$

Just as before, our general model is not expressed entirely through the graph, and an ideographic example is specified by providing table of coefficient values. Our table is now quite a bit larger, but these coefficients have meaningful definitions which relate to our theory. While this formulation offers a huge improvement over the linear formulation, we can still imagine relationships which this formulation cannot express. Thus, a graph-wide assumption that each edge represents a differential equation may not be general enough for our HBM specification.

It should be noted at this point that although the linear formulation is too simple to express the dynamics of the differential formulation, the differential formulation is capable of expressing linear relationships. This is accomplished by setting coefficients of dynamical components to 0. One might think then that there is some general formula which could express any functional form and that this form should be used to express the relationships between variables in our HBM graphs. While such formulations do exist (such as Taylor or Fourier series approximations or even ANN-based relations), this usage tends to make the model difficult to understand and to simulate with. Indeed, linear and differential formulations are in such widespread use because of the relative ease with which we can understand and solve them. An additional problem raised by this approach is that of redundant formulations. Indeed formulations could even subvert the "arrow direction" by drawing information contrary to our original choice of notation. The graph can then no longer be considered a directed graph, and the meaning of an edge becomes less clear. Additionally, the table of coefficients needed to express an ideographic case of the model quickly becomes prohibitively large, and the effect of each coefficient on the outcome is not intuitively meaningful.

Let us now consider the case where a graph-wide assumption is NOT made. That is, we will specify the functional form of each node individually so that each edge on the graph may be linear in form while another may be differential. This has the benefit of allowing for both complex relationships between variables as well as simplistic ones. In this way one could craft a model in which two variables are linearly related and a third is dependent on the variance of another variable (a particularly odd formulation, but one which is relevant to behavioral theory). Unfortunately, this approach also means that a table of formulations must now be included with our graph to show the meaning of each edge in the graph. Consider for example the table below for Graph 2:

node	formulation	type
B	$coeff_ab * A(t) + coeff_cb * C(t) + cosnst$	linear eq.
D	$TODO$	differential eq.

Table 1: an example formulation table for Graph 2

If a fixed number of functional forms is adhered to, the graph can be made to visually represent these functional forms through the use of different node icon shapes. This approach quickly begins to resemble applications which use flow-based programming. Indeed, they are quite similar in their approach, and the specification of a HBM is quite similar to the writing of a program.

Note also that a node can set any arbitrary formulation if needed. This is the least desirable situation, since the meaning of an inflow to each node even more convoluted. Though this usage reduces the ability of the graph to reduce processing load on the user through abstraction, there may be some cases where this type of formulation is desirable. Consider the following graph and formulation table:

$$A => B <= C => D => E$$

Graph 3: The final example graph.

node	formulation	type
B	$coeff_ab * A(t) + coeff_cb * sqrt(C(t)) + const_b$	custom
D	$TODO$	differential
E	$coeff_de * D(t) + const_de$	linear eq.

Table 2: an example formulation table for Graph 3

In conclusion, we propose that an HBM should be specified using the following rules:

1) use a graph-wide formula assumption if possible
2) when choosing a formulation, consistency between nodes is most important
3) when choosing a formulation, simplicity and clarity is second only to consistency
4) Thus a HBM specification must include:
    a) An information flow graph
    b) one of:
        - a graph-wide formulaic assumption
        - a formulation table

In order to express a particular solution of a general HBM, an ideographic HBM must also include a table of coefficient values.

B. Tool Design Guidelines

The guidelines provided below are generalizable findings from our studies which may be useful for other developing modeling and simulation tools for behavioral scientists.

Ease of use and intuitive user interface is a primary design consideration for existing softwares for modeling and simulation in systems theory. (reference and cite vensim, etc, from here) The ease of use of a system is one of the most important factors in the adoption of the system. (cite technology acceptance model?) The trans-disciplinary nature

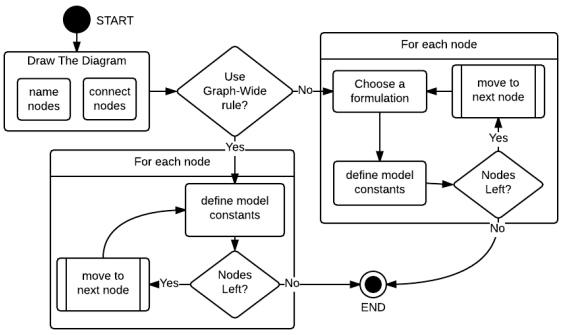

Fig. 3. Flow chart depicting the process of building a graphical HBM.

of a human-behavior modeling toolkit may require special consideration in order to be usable by those who may be unfamiliar with modeling and simulation concepts.

TODO: Figure: path2flow.png example of the rise in model complexity for the theory of planned behavior when mathematical assumptions are made explicit. (from: Ajzen 1985,1991,2002 (left), Nandola et al. 2013 (right))

(overview and definitions)

* single-page design

* information-searching / focus and context design

* walk-through first use instead of tutorial (see, copy, do) /cite?

* use existing terminology (but also define it)

* a summary of lessons learned and suggestions for future works

* multipage vs single-page (v0.1-¿v0.2)

* inquiries in panel-reviews

* single-page is overwhelming

* multiple terminologies from user surveys /citeontology customizable?

IV. Proposed Tool: Model-Builder v3

* noflo, flow-based programming style * how does this new design use the design guidelines outlined?

A. User Interface Design for HBM-builder

The flow chart shown in figure **??** represents the movement of a user through the process of building an HBM, or any graphical model for that matter.

It is important to note that the process of building is pseudo-linear; users will likely want to jump back to an arbitrary point in the process and as the model takes shape. As the user moves through the process, they will re-assess the meaning of variables and connections and their model will solidify. UI for an HBM-building tool should enable easy backtracking and should work to provide feedback on the characteristics of the model early.

The second generation of the Model Builder Tool presents a guided process which takes place within a single-page application. In this way, users can see how choices influence the model in real-time, and thus can iterate on their design much sooner.

While keeping the non-linearity of this process in mind, we will now walk through the diagram and address design considerations for each part of the process.

1) Draw the Diagram: This task involves the creation and connecting of all nodes in the diagram. Nodes and their connections should be primarily based in theory, so a user should be able to build a first draft of their model with little feedback. Once the formulas for each node have been specified, however, it is likely that the user will want to return to this step and reassess the network. Several paradigms diagram-drawing have been well tested, but there are essentially two prevailing options: 1) drag-and-drop node creation with drag-to-connect edges, and 2) textual “diagram specification language”.

a) drag-and-connect: pros:

1) easy and intuitive to use

cons:

1) can be difficult to position nodes to reduce edge overlap
2) users often distracted trying to make graph ”pretty”
3) requires constant switching between mouse and keyboard2

b) Diagram Specification Language: pros:

1) can be much faster than mouse-based
2) searching/finding nodes for modification is as easy as textual search

cons:

1) parsing errors can be a source of confusion
2) user must learn the markup ”language”

2) Use Graph-Wide Rule?: This choice can come before or after the initial drawing of the graph, since it is likely the user has a graph-wide assumption in mind when starting the process. This choice determines whether or not the interface for specifying nodes individually is shown, so it is logical to think of the ”none”, ”node-specific”, or ”node-unique” option as one of the choices available here.

As an example, some graph-wide rules which may be implemented include:

1) Probabilistic Graphical Network
2) Linear Sum
3) Fluid-Flow Analogy / Differential Equation Formulation (1st order and 2nd order)

If a graph-wide rule is set, the formulation section for each node should show as locked to the rule’s respective formulation. Editing an individual node’s formulation will disable the graph-wide rule, and should require confirmation from the user.

The interface for choosing one of these rules may be as simple as a select-option box or may include a detailed rule description and example.

An additional feature which may be useful is to allow for user-defined graph-wide rules. In this case, users might choose "create new rule" and an interface to allow users to edit and save graph-wide rules is needed.

3) For Each Node: Items encapsulated by the "for each node" swim-lanes must be executed for each node in the graph. The graph-wide-rule-yes case differs from the graph-wide-rule-no case only in that the graph-wide-rule-no case requires the additional step of setting the formulation at each node.

4) Choose a Formulation: This item is identical to the graph-wide rule choice, except that it applies only to a single node. Similar design considerations apply, and user-defined choice is even more important at this scale.

5) Define Model Constants: The general solution of an HBM does not require definition of the constants, but a simulation cannot be run until some numerical value is assumed. These constants often have theoretical significance in that they often have meaningful influence upon system behavior. Scaling-coefficients, for instance allow for relative weighting of each inflow. Similarly, the coefficients of a dynamical equation define how quickly variables react to a change "upstream".

UI for setting these constants could be as simple as a numerical input for each coefficient's name, but additional model details provided here may be of great benefit to the model designer.

Firstly, UI should include equation-specific descriptions of the constants whenever possible. This means that a separate UI for each formulation type is necessary.

Secondly, adjustment of coefficients would ideally show changes in the dynamics of a node's value in real-time. This can take the form of a time-series chart of the selected node's numerical value over time along with (editable) time-series for the inflows. Adjustment of these inflow series may also take a number of different forms.

Thirdly, model creators may want to set these coefficients in three different ways:

a) 1) Set the coefficient value to a constant: All simulations derived from this general solution will have this exact value.

b) 2) Set a probability distribution for the coefficient: Simulations derived from this general solution will select a numerical value based on the given distribution. This distribution may be learned from a training dataset, or bounds may be set for later training.

c) 3) Leave the variable unbounded and assume a flat probability distribution: This selection is identical to option 2 with a flat distribution from -infinity to +infinity.

These different ways of defining coefficients in a HBM become very important when it is time to run simulations and compare model results to real data.

Lastly, model creators may want to use a measured value to set the value of these coefficients. In this way, one measured input (in real life), such as big-5 personality type, can be used to set the value of multiple coefficients across different variables in the model.

6) Nodes Left?: No if there are un-specified nodes remaining, yes if all are complete. It should be left to the user to confirm model-completeness, however, since a great deal of model-tweaking is likely now that simulation results can be displayed.

7) Next Node: Progression downstream through the model should be relatively straightforward in the case of tree-like models, but feedback loops can cause issues here. Perhaps the best paradigm here is to progress as naturally as possible, but allow the user to over-ride and select any node. This interaction mode is needed to allow for later model-tweaking anyway.

The behaviorSim Model Builder v0.2 currently does not include significant enough indicator of node "completeness", so that the user is sometimes unsure when they should feel free to move to the next node.

V. Conclusion

We have provided design guidelines to aid in development of modeling and simulation tools for behavioral scientists based on lessons learned in the development process, and have outlined the design of an application which utilizes these guidelines.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.