

Explaining the Behavior of System Dynamics Models

MÁRCIO DE OLIVEIRA BARROS
CLÁUDIA MARIA LIMA WERNER
GUILHERME HORTA TRAVASSOS

COPPE / UFRJ – Computer Science Department
Caixa Postal: 68511 - CEP 21945-970 - Rio de Janeiro – RJ
Voice: 5521 562-8675 / Fax: 5521 562-8676
{marcio, werner, ght}@cos.ufrj.br

Abstract

In this paper we present a technique that helps the analysis of system dynamics models. The technique, namely *Event Tracking*, maps simulation trends over time to predefined events. It uses state machines whose behavior can be traced to changes suffered by selected variables in a system dynamics model. Changes to variable values trigger messages, which are presented to the model analyst to help the interpretation of the underlying model behavior.

Event Tracking is an interesting feature to system dynamics simulators, since it maps the mathematical results achieved through simulation to natural language statements. It allows a trainer assistant to define the relevant events for a model, highlighting the model major features through event messages. We have implemented the proposed technique in the ILLIUM system dynamics simulator. The simulator also allows a student to track model behavior, executing a simulation and following the presented messages.

KEYWORDS: model analysis, event tracking, simulator tools

1 Motivation

The behavior of large system dynamics models is usually hard to understand. When a behavior equation is composed by several variables, each one described by a complex equation, the evaluation of causal relationships among model parameters and results requires the analysis of a large set of simulation trends. This task is even more difficult when feedback loops are present, due to the recursive effect of some variables upon their future behavior.

The mathematical representation of system dynamics models is interesting to formally describe the rules embedded within the model. However, this representation is not well suited for human interpretation. Therefore a technique to map the mathematical results achieved through simulation to a representation better suited for human analysis is worthwhile.

We observed the need for such a technique during a software engineering graduate course. At this course, we have asked a group of students to analyze the software project system dynamics model developed by Abdel-Hamid and Madnick (1991). This model captures the effects of management decisions upon the cost and development time of a software project, being composed by more than two hundred equations. The students were supposed to run some predefined simulations, each one describing a particular management decision, and determine the impact of such decisions upon the software project behavior. They were successful in running the simulations, but they could not interpret their results without the aid of supervisors who have been studying the model for several weeks.

Since we believe that system dynamics models are valuable training tools and should be analyzed without on-site help of specialists, we observed the way some analysts use to interpret model results in order to create a technique that students can apply to this end. We

perceived that during the analysis of a model, analysts track the values of selected variables and extract the relevant events that occurred during model execution from their simulation trends. These events are used to explain model behavior.

In this paper we present a technique that automates the tracking of simulation trends. The technique searches for predefined behavior patterns, identified by specific configurations of variable values, and triggers messages to alert the user about the occurrence of relevant events during the simulation. These messages translate the simulation results to natural language. The technique, namely *Event Tracking*, allows the specification of the relevant events and their tracking along simulation trends.

This paper is organized in four sections. Within this one we present the motivation for this work. In Section 2, we present the concept of event sets and their composition. In Section 3, we present the current implementation of the *Event Tracking* technique and an example of its application. Finally, Section 4 presents the future perspectives of this work, highlighting the need for experiments that prove its usefulness.

2 Event Sets

Event Tracking is based on executing state machines whose behavior is dictated by the values of predefined model variables. An *event set* is a state machine that maps particular values of one or more variables to events that are relevant to the interpretation of a model behavior. As any state machine, an event set is composed by states and transitions.

A *state* represents a set of characteristics that can be found on an event set in a simulation step. The execution of an event set starts from a specific state, namely the *initial state*. It proceeds as the event set changes states due to the activation of its transitions. There can be only one initial state in an event set.

A *transition* is a directed relationship between two states. A state change occurs when a transition originating from the current state is activated. Every transition has an associated condition that depends on a model variable. The condition specifies the range of values that the variable must reach to activate the transition. In every simulation step, the transitions originating from the current state are evaluated and, if a transition is activated, the current state is set to its target state.

Every transition is also associated with a message. When the transition fires, the message is sent to the user. Event Tracking logs these messages, which are supposed to highlight events that help the user to understand model behavior. Transition messages can be parameterized by model variables. Such parameters are replaced by the variable values in the particular simulation step when the transition was activated.

Transition conditions can also account for the difference between the current value of a model variable and its value when the event set reached the current state for the last time. The condition can specify the range of values that this difference must reach to trigger the transition, instead of specifying a range for the variable value. When an event set enters a state, the values of the variables associated to its transitions are saved for future evaluation within these conditions.

Several event sets can be concurrently analyzed during a single simulation. Event sets are defined by model specialists and made available to less experienced model analysts. The specialists are supposed to identify the most relevant variables within a model and the events that explain model behavior. Event sets map these events, being distributed with the model and ran by its analysts.

3 The Current Implementation

Event Tracking is currently implemented within the ILLIUM system dynamics simulator (Barros, 2001). The simulator was developed at COPPE/UFRJ to allow the research of techniques and tools to support system dynamics modeling and simulation. The simulator has event set execution capabilities, logging transition messages and presenting these messages to the user. Figure 1 presents a snapshot from the Event Tracking tool of the ILLIUM simulator. The tool shows the available event sets in the left and the messages captured in the last simulation in the right. The number of simulation steps is also presented in the top-left corner of the window.

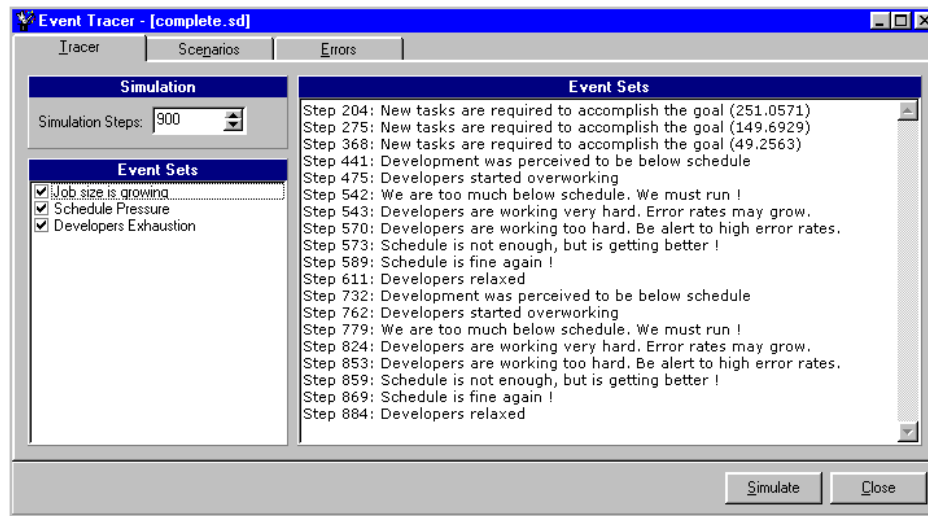


Figure 1 – Event Tracking tool implementation in the ILLIUM simulator

The event sets in Figure 1 were developed for the Abdel-Hamid and Madnick (1991) software project model. They capture the relationships of a growing job, schedule pressure, developers overwork, and error rates. Model analysts can observe that, as work grows and the conclusion date is not modified, schedule pressure raises. As developers work harder to accomplish more work in the same schedule, lower quality code is produced and error rates rise. Event set are distributed in a XML file format (W3C Consortium, 2000). Since the current implementation of the ILLIUM tool does not provide an event set authoring tool, XML was selected due to the availability of XML editors. Figure 2 presents the description of the schedule pressure event set.

```
<EVENTSET name="Developers Exhaustion">
  <STATE name="Base" Initial="True">
    <TRANSITION target="Overworking" variable="AFMDPJ" condition="GE 0.7">
      Step $Step: Developers started overworking</TRANSITION>
    </STATE>

  <STATE name="Overworking">
    <TRANSITION target="Hardwork" variable="AFMDPJ" condition="GE 0.9">
      Step $Step: Developers are working very hard. Error rates may grow</TRANSITION>
    <TRANSITION target="Base" variable="AFMDPJ" condition="LT 0.7">
      Step $Step: Developers relaxed</TRANSITION>
    </STATE>

  <STATE name="HardWork">
    <TRANSITION target="Slavery" variable="AFMDPJ" condition="GE 1.1">
      Step $Step: Developers are working too hard. Be alert to high error rates.</TRANSITION>
  </STATE>
</EVENTSET>
```

```
<TRANSITION target="Base" variable="AFMDPJ" condition="LT 0.7">
  Step $Step: Developers relaxed</TRANSITION>
</STATE>

<STATE name="Slavery">
  <TRANSITION target="Base" variable="AFMDPJ" condition="LT 0.7">
    Step $Step: Developers relaxed</TRANSITION>
  </STATE>
</EVENTSET>
```

Figure 2 – XML representation for the *Developers Exhaustion* event set

4 On Going Work and Future Perspectives

In this paper, we presented *Event Tracking*, a technique that uses state machines whose behavior is driven by model variables. They map mathematical results achieved through simulation to natural language, which is more suitable to model behavior interpretation.

From our first observations, Event Tracking seems a promising technique. However, we have not yet made experiments to prove its usefulness. We intend to plan and run an experiment to verify if the proposed technique helps the interpretation of simulation results. We also intend to improve its current implementation developing an event set authoring tool. We expect to publish results obtained from the use of Event Tracking in future papers.

Acknowledgements

The authors would like to thank CNPq, CAPES and FINEP for their financial investment in this work.

References

- Abdel-Hamid, T., Madnick, S.E. (1991). *Software Project Dynamics: an Integrated Approach*, Prentice-Hall Software Series, Englewood Cliffs, New Jersey
- Barros, M.O. (2001) *ILLIUM - System Dynamics Simulator*, ILLIUM tool homepage at URL <http://www.cos.ufrj.br/~marcio/Illium.html>
- W3C Consortium (2000). "Extensible Markup Language (XML) 1.0 (Second Edition)", W3C Recommendation, available at <http://www.w3.org/TR/REC-xml>