Traviando - Debugging Simulation Traces with Message Sequence Charts *

Peter Kemper
Department of Computer Science
College of William and Mary
Williamsburg, Virginia 23187-8795, USA
kemper@cs.wm.edu

Carsten Tepper
Informatik IV
Universität Dortmund
D-44221 Dortmund, Germany
carsten.tepper@udo.edu

1 Motivation

Discrete event simulation is broadly applied in the analysis of dynamic and concurrent systems and often with complex and detailed models. The process of creation and coding a non-trivial simulation model is error prone and the identification and understanding of what really goes on in a simulation run needs adequate tool support for makeing verification, validation and debugging productive. This holds in particular for the analysis of simulation traces, a powerfull debugging aid with the challenge of large amounts of data. In this paper, we present Traviando, a Trace Visualizer and Analyzer from Dortmund university.

2 Traviando: Tool Description

Traviando is a standalone Java implementation that reads a trace of a discrete event system and provides visualization and analysis techniques to investigate the dynamic behavior as documented in that trace. A discrete event system is seen as a finite set of interacting processes that perform local events or perform events that describe a synchronous or asynchronous interaction among several processes. In this section, we briefly discuss Traviando's input, output and analysis techniques.

Input. Traviando reads a trace from a file in a XML format, where a trace consists of a header and a sequence of events (or a sequence of states and events). The header is used to declare which processes, state variables and events exist, their attributes, their valid values and how variables and events are associated with processes. Hence, the header provides valuable structural information and also helps to keep the description of the sequence of events more concise. The sequence of events carries information on the order of events. Optional additional information like changes of state and time stamps are relevant for many analysis features.

The trace concept and XML format have been successfully employed to use Traviando for the analysis of simulation models generated by three different modeling frameworks so far, namely the ProC/B toolset [1], the APNN toolbox [2], and Mobius [3]. The ProC/B toolset is a modeling framework for logistic models that follows the common process-interaction approach for discrete event simulation modeling. The APNN toolbox is a Petri net oriented modeling framework that provides qualitative analyis based on model checking as well as discrete event simulation of timed models and Markov chain analysis for quantitative analysis. Mobius is a multi-formalism multi-solution framework developed by W.H. Sanders at the University of Illinois at Urbana Champaign. It is widely used in industry and supports discrete event simulation as well as numerical analysis based on Markov chains.

Output, Visualization. Traviando visualizes a trace as a message sequence chart. Fig. 1 illustrates the concept with a screen shot of a small example of 6 processes. Traviando offers a number of features for visualizing large amounts of data. For instance, subsets of processes can be grouped which helps a user to adjust their focus. Colors and highlighting are used in many ways to identify and distinguish among events. The particular visualization features are presented in [5] in some further detail.

Quantitative Analysis. Traviando helps to track down events that effect performance measures that are estimated in a simulation run. In particular for ProC/B models, events are paired as call and return of a function, resp. service call. For those pairs, Traviando derives empirical distributions of response times at processes, statistics on the numbers of open calls, and uses that information to visualize extremal behavior and to highlight fragments of a trace that are extraordinary and thus should attract the attention of a modeller. A more detailed discussion on Traviando's analysis features for tracking down performance problems is presented in [4].

Qualitative Analysis. Traviando provides a linear time logic (LTL) model checker, so a modeller can specify properties of interest as LTL formulas and Traviando checks and highlights states, resp. events, of a trace that have that prop-

*This material is based upon work partly funded by Deutsche Forschungsgemeinschaft, SFB 559.

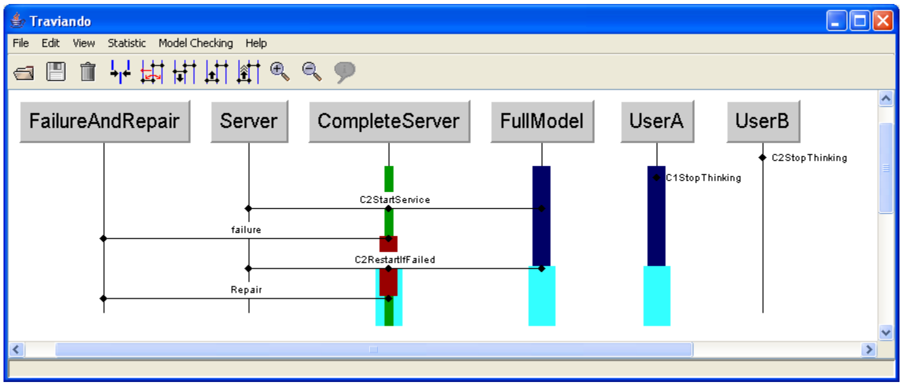

Figure 1. MSC of Mobius Example Trace

erty. We implemented a graphical formula editor based on the pattern system of Dwyer et al to make the specification of properties by logic formulas more productive and reliable even for a non-expert user. The colorings in Fig. 1 indicate where particular formulas, subformulas and atomic propositions hold for that example. Atomic propositions are boolean expression built from arithmetic expressions on state variables. The formula editor gives a graphical, tree-type representation with one node per logical operator and atomic propositions as leafs. Formulas can be instantiated from particular predefined patterns and refined in various manners. Atomic propositions can be selected by mouse-clicks from a menu that is derived from a currently defined set atomic propositions. A set of atomic propositions is defined in an additional editor window. Colors are associated with nodes in the formula editor to define the coloring and highlighting used for the visualization of the model checking results on a MSC. For further details, we refer to [6]. For the rest of the paper, we highlight a particular innovative feature.

Cycle Detection and Reduction. A trace gives a sequence of states and events in a total order. Certain states may occur multiple times. If that is the case, we denote this as a cycle, which describes a repetitive behavior. If we are interested on a how a particular state or fragment of a trace can be reached, we can remove cycles from the prefix that leads to the fragment of interest. Algorithms for computing cycles in a sequence of states are known, resp. simple to derive. What we identified as crucial and non-trivial is to select an appropriate subset of non-overlapping cycles that gives a maximal reduction of the prefix. In [6], we propose 3 heuristic strategies for the selection of cycles and evaluate the overall effect on an example trace of a Möbius simulation model. Indeed, Fig. 1 shows the benefits of the cycle reduction for a trace of a reliability model with several thousands of events: the resulting trace that leads to the fragment of interest is in the order of 10 events. Cycle reduction helps to extract those events that are relevant to reach a fragment of interest.

For further information, documentation, availability visit http://www4.cs.uni-dortmund.de/Traviando.

References

[1] F. Bause, H. Beilner, M. Fischer, P. Kemper, and M. Völker. The ProC/B toolset for the modelling and analysis of process chains. In T. Field et al, editors, *Computer Performance Evaluation (TOOLS'02)*, Springer LNCS 2324, pages 51–70, 2002.

[2] Falko Bause, Peter Buchholz, and Peter Kemper. A toolbox for functional and quantitative analysis of DEDS. In Ramón Puigjaner et al, editors, *Computer Performance Evaluation (TOOLS'98)*, Springer LNCS 1469, pages 356–359, 1998.

[3] Tod Courtney, David Daly, Salem Derisavi, Shravan Gaonkar, Mark Griffith, Vinh Vi Lam, and William H. Sanders. The Möbius modeling environment: Recent developments. In Proc. *QEST*, pages 328–329. IEEE Computer Society, 2004.

[4] P. Kemper and C.Tepper. Trace based analysis of process interaction models. In *Proc. of the 2005 Winter Simulation Conference*, pages 427–436, 2005.

[5] P. Kemper and C. Tepper. Visualizing the dynamic behavior of ProC/B models. In T. Schulze et al., editor, *Simulation und Visualisierung*, pages 63–74. SCS Publishing House, 2005.

[6] P. Kemper and C. Tepper. Trace analysis - gain insight through modelchecking and cycle reduction. Technical report, Universität Dortmund, SFB 559 Modelling Large Networks in Logistics, No 06007, ISSN 1612-1376, 2006.