ENHANCEMENTS OF STATECHART-MODELING— THE KIEL ENVIRONMENT

Steffen Prochnow and Reinhard von Hanxleden
{spr, rvh}@informatik.uni-kiel.de
Dept. of Computer Science, Christian-Albrechts-Universität Kiel
http://www.informatik.uni-kiel.de/rtsys/

Abstract

The Kiel Integrated Environment for Layout (KIEL) is a prototypical modeling tool to explore novel editing, browsing and simulation paradigms in the design of complex reactive systems.

1. Introduction

Modeling systems based on semi-formal graphical formalisms, such as Statecharts [5], has become standard practice in the design of reactive embedded devices. However, the modeling of realistic applications often results in very large and unmanageable graphics, severely compromising their readability and practical use. To overcome this, we present a methodology to support the easy development and understanding of complex Statecharts.

2. The KIEL Environment

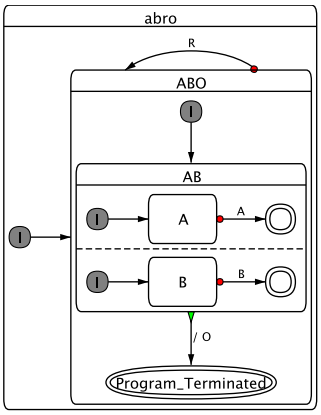
The *Kiel Integrated Environment for Layout* (*KIEL*) environment [11] is a prototypical modeling tool to explore novel editing, browsing and simulation paradigms in the design of complex reactive systems. *KIEL* is not restricted to a specific Statechart dialect; so far, it has been adapted to SyncCharts/Safe State Machines (SSMs) [1], Stateflow [12] and UML-Statecharts [6].

Statechart Layout: A central enabling capability of *KIEL* is the automatic layout of Statecharts, which computes bottom-up layouts at each hierarchy level using *GraphViz* [4]. This transforms any given Statechart to a standardized layout (*Statechart Normal Form*, SNF) that is compact and makes systematic use of *secondary notations* to aid readability.

Statechart Editing: As an alternative to the classic, low-level WYSIWYG graphical editing paradigm, *KIEL* provides a *structure-based graphical editor*, which applies high-level editing commands (*e. g.*, "add successor state") to

```
module ABRO:
input A, B, R;
output O;
loop
  [ await A || await
        B ];
  emit O;
each R
end module
```

(a) Esterel



(b) Safe State Machine

```
statechart abro[model="Esterel Studio";version="5.0"]{
 input A;
 input B;
 input R;
 output O;
 {
  ->ABO;
  ABO{
   AB{
     ->A;
     A->AF[type=sa;label="A"];
     AF[type=final];
     ||
     ->B;
     B->BF[type=sa;label="B"];
     BF[type=final];
   };
   ->AB;
   AB->Program_Terminated[type=nt;label="/ O"];
   Program_Terminated[type=final];
  };
  ABO->ABO[type=sa;label="R"];
 };
};
```

(c) Textual Description Language

Figure 1: Different Representations of an SUD Example [1]

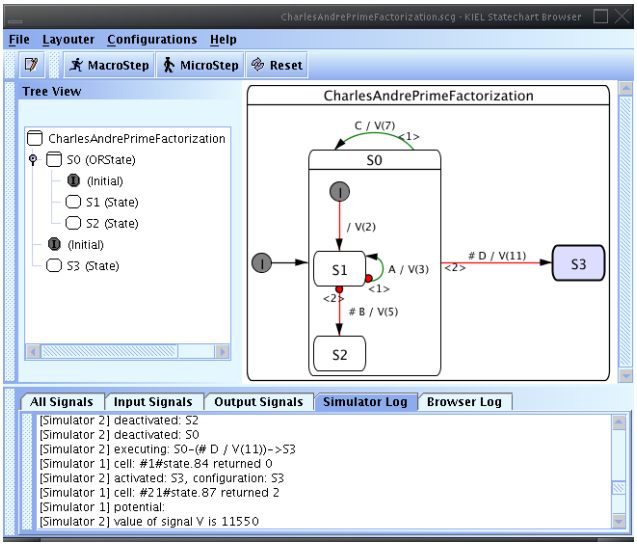


Figure 2: Screenshot of KIEL as it simulates an SSM

a model-under-development. As another alternative, a designer can edit a textual description (see Figure 1c), while the graphical representation is updated automatically.

Statechart Import: In addition to the internal editors, *KIEL* can import Statechart models from external tools. Currently supported are Esterel-Studio [3], Matlab/Simulink/Stateflow, and ArgoUML [2].

Statechart Synthesis: *KIEL* provides an approach to synthesize SSMs from (textual) Esterel v5 programs [8]. This permits a design flow where the designer develops a system at the Esterel level, but uses a graphical browser and simulator to inspect and validate the system under development. The Figures 1a and 1b show an Esterel SUD and the equivalent SSM representation.

Simulation: A common problem when simulating complex systems with many concurrent activities is that designers easily loose track of the current overall system state. *KIEL* addresses this by a "dynamic semantic focus-and-context representation," which provides different *views* of the system depending on the system state [10]. Figure 2 presents a screen-shot of *KIEL* as it simulates an *Safe State Machine* (including an automatically layouted SSM).

Style Checking: Ruling out certain modeling constructs helps to avoid common types of errors. *KIEL* provides an automated checking framework, which checks compliance to *robustness rules* [7]. The framework provides a wide range of pre-defined dialect-dependent rules, and also allows to express new design rules in OCL, or in Java. The latter is also used to incorporate a theorem prover for more advanced checks, such as determinism.

3. Summary and Future Work

KIEL's provides novel editing, browsing and simulation paradigms to enhance the comprehension of the system under development. The feedback we have obtained so far regarding the concepts of editing, the SNFs, and dynamic Statecharts has been quite positive. This has also been supported by an empiric study [9].

Regarding ongoing and future work, there are numerous ways in which to extend the capabilities of *KIEL*. Currently, we are focussing on the extension to data-flow diagrams, similar to SCADE or Simulink; we are also considering how to integrate the *KIEL* capabilities into generic modeling frameworks, such as the Eclipse IDE.

References

[1] C. André. Semantics of S.S.M (Safe State Machine). Technical report, Esterel Technologies, Sophia-Antipolis, France, Apr. 2003. available at http://www.esterel-technologies.com, in the download section.

[2] ArgoUML. Tigris.org: Open Source Software Engineering Tools. http://argouml.tigris.org/.

[3] Esterel Technologies. Esterel studio. http://www.esterel-technologies.com/products/esterel-studio/overview.html.

[4] GraphViz. Graphviz—graph drawing tools. http://graphviz.org/.

[5] D. Harel. Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, 8(3):231–274, June 1987.

[6] Object Management Group. Unified Modeling Language: Superstructure, Version 2.0, Aug 2005.

[7] S. Prochnow, G. Schaefer, K. Bell, and R. von Hanxleden. Analyzing Robustness of UML State Machines. In *Proceedings of the Workshop on Modeling and Analysis of Real-Time and Embedded Systems (MARTES'06), held in conjunction with the 9th International Conference on Model Driven Engineering Languages and Systems, MoDELS/UML 2006*, Genua, Oct. 2006.

[8] S. Prochnow, C. Traulsen, and R. von Hanxleden. Synthesizing Safe State Machines from Esterel. In *Proceedings of ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES'06)*, Ottawa, Canada, June 2006.

[9] S. Prochnow and R. von Hanxleden. Statechart Development Beyond WYSIWYG. In preparation.

[10] S. Prochnow and R. von Hanxleden. Comfortable Modeling of Complex Reactive Systems. In *Proceedings of Design, Automation and Test in Europe (DATE'06)*, Munich, Mar. 2006.

[11] The KIEL Project (Kiel Integrated Environment for Layout). Project Homepage, 2006. http://www.informatik.uni-kiel.de/rtsys/kiel/.

[12] The Mathworks. Stateflow—Design and simulate event-driven systems. http://www.mathworks.com/products/stateflow/.