Enhancements of Statechart Modeling—The KIEL Environment

Steffen Prochnow and Reinhard von Hanxleden

Department of Computer Science and Applied Mathematics
Real-Time Systems and Embedded Systems Group
University of Kiel, Germany

ARTIST'07, July 2007

Introduction

Motivation:

- Statecharts possess high complexity (combinations of components, dependencies, system dynamics, concurrency)
- tools for modeling Statecharts provide restricted facilities to enter and understand complex system behavior

Purpose:

- formulation of improvements for easy modeling, analyzing and understanding complex Statecharts
- establishment of these improvements in a highly configurable tool for modeling and simulation
- validation of operativeness of the tool

Layouting Statecharts

Kiel **I**ntegrated **E**nvironment for **L**ayout

- uses several layout heuristics to choose from
  - a simple horizontal/vertical layout scheme
  - more advanced schemes, provided by GraphViz
- provides generic wrapper to create hierarchical layout from flat layout schemes
- implemented in Java
- highly configurable

Creating Statecharts

Approaches:

1. quick-and-dirty graphical model (WYSIWYG)
   - import from Esterel Studio, Matlab/Simulink/Stateflow
2. structure-based editing
   - selection and manipulation (KIEL-Macro editor)
   - Statechart production rules
3. textual languages
   - KIT (Statechart description language)
   - Esterel

Creating Statecharts

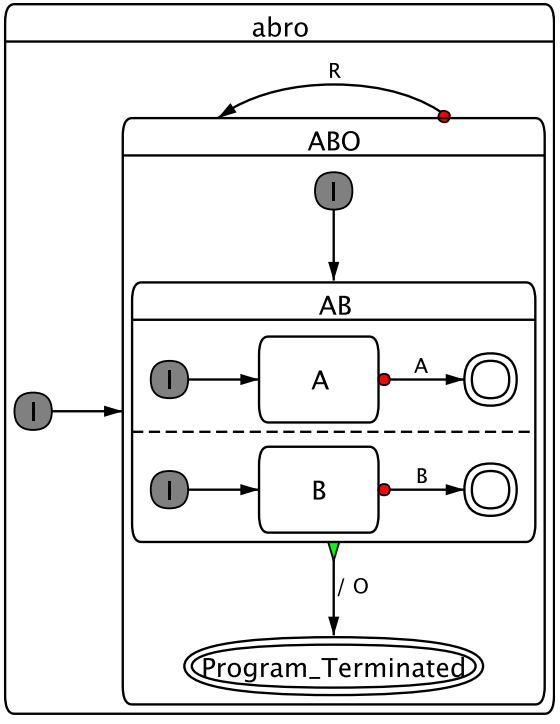
- Different Representations of an SUD Example

```
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"];
 };
};
```

(a) KIT—Textual Description Language

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

(b) Esterel



(c) Safe State Machine

Visualizing Statecharts

Idea: Views should hide in-active sub-states

- present dynamically changing views dependent on
  1. simulation state
  2. user requests
- a dynamic extension to semantic focus-and-context representation (Köth)
- Views:
  - associated with deepest hierarchy levels of macro states
  - all simple states of this level share one view
  - each view shows complete system

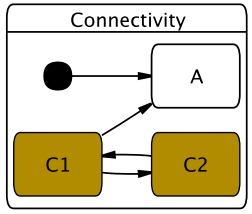
Checking Statechart Style

Statechart Style Guide:

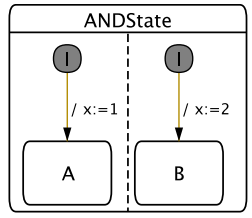
- operational instructions for humans and configuration for automated analysis
- set of 41 wellformedness-, syntactic, and semantic rules
- defines a subset of the language Statechart

Statechart Style Checking:

- based on defined Style Guide
- allows to express new rules in OCL or in Java
- theorem prover for more advanced checks



Connectivity



Race Conditions

Syntactic Rules

Dwelling

Semantic Rules

Summary and Outlook

The KIEL Prototype (Summary)

- automatic layout of Statecharts
- several layout heuristics
- interfaces to Esterel Studio and Stateflow
- supports dynamic Statecharts
- easy textual modeling
- transformation of Esterel to SSM
- checking of syntactical/semantical properties
- has been used successfully in teaching "System Modeling and Synchronous Languages"
- empirical experiment evaluation shows efficiency and practicability
- URL: `http://rtsys.informatik.uni-kiel.de/~rt-kiel`

Outlook on KIEL

- examine further layout schemes
- refine secondary notations for Statecharts (et al.)
- extensive explorative analysis of the empirical study
- layout, textual description with graphical model synthesis, and simultaneous display for data-flow languages (SCADE/LUSTRE)

thanks!
questions or comments?