Synchronous Statecharts
for executing Esterel with Ptolemy

Christian Motika

Real-Time Systems and Embedded Systems Group
Department of Computer Science
Christian-Albrechts-Universität zu Kiel, Germany

DREAMS Seminar,
03. OCT 2011
Embedded Systems

- Designed for dedicated applications inside a surrounding system
- „A computer that is not perceived as such“
- Often non-terminating SW, continuous interaction w/ HW
  ⇒ Reactive Systems
- Often safety-critical systems
Reactive Systems

- Continuous, non-terminating interaction
- Pace is controlled by the environment
- Computation (output) of steps of the reactive system ⇒ Reaction vs. interaction
- When to produce outputs? In real-time! → WCRT
Synchronous Languages

- Separate concerns
  1. Functionality
  2. Timing
- Specify functionality, logical timing only (order)
- Goal: Fully deterministic behavior
- Assume zero reaction time $\Rightarrow$ perfect synchrony w/ interaction

[Diagram showing synchronous statechart elements: signals emitted by the environment (input), synchrony hypothesis ("global clock"), cycle start, cycle end, sequences of transitions, emitted signals (output), single-cycle reaction.]

[G. Luettgen 2001]
Motivation

- Synchronous model of computation (MoC):
  - Esterel, SyncCharts, SC (control flow)
  - Lustre, Signal, SCADE (data flow)
  - Ptolemy (SR domain)

- SyncCharts a synchronous statechart dialect
  - Primary example for KIELER framework

- KlePto: Executing SyncCharts w/ Ptolemy

- KIES: Esterel to SyncCharts transformation
  - Execute Esterel w/ Ptolemy
Overview

- Esterel
- SyncCharts
- SyncCharts Execution (KlePto)
- Esterel to SyncCharts transformation (KIES)
  - Demo
- Summary
What is Esterel?

▶ Synchronous language
▶ Imperative
▶ Control flow oriented
▶ Deterministic programs
▶ History

▶ Developed by J.-P. Marmorat and J.-P. Rigault (at Ecole des Mines de Paris)
▶ G. Berry developed a formal semantics for Esterel in 1983
▶ Esterel v5: Has been stable since late 1990s
▶ Esterel v7: same principles as in v5, several extensions
Synchrony in Esterel

**Definition** [Perfect Synchrony]
A system works in **perfect synchrony**, if all reactions of the system are executed in zero time. Hence, outputs are generated at the same time, when the inputs are read.

- In practice, ‘zero time’ means before the next interaction
- Macro steps consist of only **finitely** many micro steps
  - ⇔ No data dependent loops in a macro step
- WCET (high-level, low-level) to meet real-time constraints
Basic Concepts

- Imperative language
  - Statement (mostly instantaneous)
  - Sequence of statements is a statement $\rightarrow s_1 ; s_2$

- Control flow oriented $\rightarrow$ pause statement

- Modules
  - Esterel programs are a list of modules
  - Module body is a statement

- Signals
  - Communication
  - Emission, Test $\rightarrow$ emit S, present S, await S

- Concurrency $\rightarrow [ s_1 \mid\mid s_2 \mid\mid \ldots \mid\mid s_n ]$

- Preemption $\rightarrow$ [weak] abort S when S
Signals

**Definition** [Signal Coherence]
A signal is either present or absent within a macro tick but never both.

- Present iff emitted, absent otherwise
  - "iff emitted" means iff an `emit` statement **must** be executed
- Constructiveness
  - Speculative executions are not allowed
  - E.g., `present S then emit S else emit S`
Example: ABRO

```
module ABRO:

input A, B, R;
output O;

loop
  abort
    [ await A || await B ];
  emit O;
  halt
  when R
end loop

end module
```

„The system has boolean valued inputs A, B, R, and an output O. Output O shall be true as soon as both inputs A and B have been true. This behavior should be restarted if R is true.“
Overview

- Esterel
- SyncCharts
- SyncCharts Execution (KlePto)
- Esterel to SyncCharts transformation (KIES)
  - Demo
- Summary
What are SyncCharts?

- Statechart dialect (Statecharts proposed by David Harel [1987])
- Built on Esterel semantics
- Invented by Charles André
- Safe State Machines
  - Was supported by the commercial tool Esterel Studio, which uses Esterel as intermediate step in code generation
SyncCharts

- Statechart dialect
- Mealy machine with
  - Parallelism, hierarchy, compound events, broadcast
- Synchrony hypothesis
  - Discrete ticks
  - Computations take no time

Charles André, Computing SyncCharts Reactions, 2003
Harel-Statecharts vs. SyncCharts—Similarities

SyncCharts are made up of elements common to most Statecharts dialects:

- States
- Initial/terminal states
- Transitions
- Signals/Events
- Hierarchy
- Modularity
- Parallelism
Harel-Statecharts vs. SyncCharts—Differences

SyncCharts differ from other implementations of Statecharts:

- Synchronous framework
- Determinism
- Compilation into backend language Esterel
- No interpretation for simulations
- No hidden behaviour
- Multiple events
- Negation of events
- No inter-level transitions
Example: ABRO

module ABRO:

input A, B, R;
output O;

loop
    abort
        [ await A || await B ];
    emit O;
    halt
when R
end loop

end module
Overview

- Esterel
- SyncCharts
- SyncCharts Execution (KlePto)
- Esterel to SyncCharts transformation (KIES)
  - Demo
- Summary
What is KIELER?

- Kiel Integrated Environment for Layout Eclipse Rich Client
- Modeling platform and test bed
  - Improve pragmatics
- Open source and Eclipse based (plug-ins)
- General concepts:
  - Generic approaches
  - Symbiosis w/ Eclipse technologies (e.g., EMF, GMF, TMF, Xpand, Xtend)
  - Interfaces to other tools (Ptolemy, Papyrus)
SyncCharts Execution in KIELER

- Esterel
- RC
- Ptolemy
- SC
- SJ

- SyncCharts
- KIES
- Xtend2
- KlePto
- Xpand2/

Christian Motika
Synchronous Statecharts
KlePto Overview

The KlePto system consists of various components that work together to transform and simulate models. The process begins with the M2M description, which is transformed into a file named dsl2moml.xtend using the dsl2moml.xtend script. This transformed model is then used to generate the model.pzo file, which contains the executable code for the Ptolemy Simulator.

The Ptolemy Simulator has two main components: the Data Producer and the Data Observer. The Data Producer is responsible for producing simulation data and model outputs, while the Data Observer is responsible for loading, executing, and observing commands and model inputs.

The Execution Manager coordinates the interaction between these components, ensuring that the simulation progresses correctly.

This system is supported by various metamodels, such as moml.ecore and dsl.ecore, and a model.dsl file, which serves as the input for the simulation.

[Semantics and Execution of Domain Specific Models, MEMWe 2010]
Ptolemy

- "The Ptolemy project studies heterogeneous modeling, simulation, and design of concurrent systems."
  
  Introduction to Ptolemy II, UC Berkeley

- Executable Models to describe behavior of reactive systems

- Ptolemy models are a set of interacting components → Actor-Oriented Design
SyncCharts in Ptolemy - Today

parallelism

Signal: \( L \)

\[ s_1 \rightarrow s_2 \]

\[ s_3 \rightarrow s_4 \]

Transitions and signals

Lo_COMBINE

SR_Director

Utilities
Directors
Actors
MoreLibraries
UserLibrary

Lo

Li

s_1_203586689

s_2_200586988

output: \( Lo = 1 \)

s_3_71121165

s_4_711121165

Lo

Li

Christian Motika
SyncCharts in Ptolemy - ModalModels and SR

- ModalModels do *cannot analysis* to expose signals that cannot be emitted
  - Signals can then be set to absent
- ModalModels can do this hierarchically
- Ptolemy models can be heterogeneous:
  - State refinement does not need to be a state machine (e.g., SR)
  - Currently limits this approach
  - Ptolemy Control Flow Domain (w/ Chris Shaver)
Overview

▶ Esterel
▶ SyncCharts
▶ SyncCharts Execution (KlePto)
▶ Esterel to SyncCharts transformation (KIES)
  ▶ Demo
▶ Summary
Model transformations

▶ Applications
  ▶ Synthesize multiple (graphical/textual) views from one model
  ▶ Edit a model (refactoring, optimization)
  ▶ Code generation
  ▶ Simulation desires

▶ Drawbacks
  ▶ Large and inflexible
  ▶ Hard to visualize
  ▶ Hard to debug
  ▶ Not interactive

▶ Goal of KIES: Address the above drawbacks
  → Use case: KIELER Esterel to SyncCharts transformation
Esterel to SyncCharts

**Motivation**

**Concept**

**Transformation Rules**

**SyncCharts Optimization**

---

Christian Motika

Synchronous Statecharts
Transformation Rule

“A transformation rule is a description of how one or more constructs in the source language can be transformed into one or more constructs in the target language” (Mens and Gorp)

- Esterel to SyncCharts
  - One rule for each Esterel statement
- SyncCharts Optimization
  - One rule for a certain SyncCharts state
- Rules presented by Lars Kühl (also formal proofs for Esterel to SyncCharts)
Implementation

Interactive Transformations for Visual Models, MEMWe 2011
Esterel to SyncCharts

**emit**

emit sig1

**loop-each**

loop s each e
SyncCharts Optimization

rule7

any

any

any

any

1

2

1

2

1

2

Christian Motika

Synchronous Statecharts
KIELER Demo

LIVE DEMO
Summary

- Research goals (long term)
  - Investigate on synchronous languages
  - Bringing together graphical and textual syntax
  - Integrate Esterel in KIELER
    - Improve pragmatics
    - Validation purposes (SC and ControlFlow Domain)

- Research goals (near term)
  - Modular and interactive transformations
    - Understand
    - Debug
  - Teaching

- Acknowledgements: Reinhard v. Hanxleden, Ulf Rüegg
To Go Further

CHARLES ANDRÉ.

GÉRARD BERRY.
The Esterel v5 Language Primer, 2000.

MOTIKA, C.
Semantics and execution of domain specific models—KlePto and an execution framework.

UC BERKELEY, EECS DEPT.
Ptolemy webpage.
http://ptolemy.eecs.berkeley.edu/.

UNI KIEL, REAL-TIME AND EMBEDDED SYSTEMS GROUP.
KIELER webpage.
http://www.informatik.uni-kiel.de/en/rtsys/kieler/.
Thank you for your attention and participation!

Any questions or suggestions?