The Synchronous Languages 12
Years Later

Tawatchai Siripanya
Seminar in Programming Language(19666)
Advisor: Lilit Hakobyan
Supervisor: Prof. Dr. Elfriede Fehr
Institute of Computer Science
Freie Universität Berlin
Date: 26.02.13
Outline

• Motivation
  • Goals of the talk
• History of the Languages
  • How the languages have been commercialized
• Distinguish of the Synchronous Languages
• Successes and Improvements
• Difficulties and Open Issues
• Summary
Motivation

Global Coordination

Source: http://www.esterel-technologies.com
Goals Of The Talk

To answer these questions:

1. Review the history of the synchronous programming languages
   - Esterel
   - Lustre
   - Signal

2. what have been achieved in the languages

3. What are difficulties in synchronous programming languages

4. What majors problems remain
Requirements of The Language

1. Concurrency
2. Simplicity
3. Synchrony

An implementation model (simple and frequently used) left (event driven), right (sample driven)

Source: [1]
The history of the synchronous programming languages

When were the synchronous languages born?

Esterel  Signal  Lustre
TIME TO THE MARKETS
Time To The Markets

Esterel  Lustre  Signal
Time To The Markets: Esterel

1980s:
A free Esterel compiler from Berry’s group at INRIA/CMA

1998:
First marketed by Simulog

1999:
Founded Esterel Technologies

2001:
Esterel Technologies brought a tool based on Lustre called SCADE (Safety Critical Application Development Environment)
Time To The Markets: Lustre

1980s:
Two big industrial Safety-Critical Software projects were born:
• The N4 Series Of Nuclear Power Plants
• The Airbus A320
But, no suitable tools available:
→ Build Own Tools
  → Airbus industries built SAO
  → Schneider Eclectrics built SAGA based on Lustre because cooperation with Lustre research group

Maintainace problems:
• Verilog undertook the problems
• SCADE Tool
Time To The Markets: Lustre

2001: Esterel Technologies brought SCADE
Time To The Markets: Signal

1990s:
• TNI owned license of Signal
• Cooperation between TNI and Snecma

1993:
• TNI developed Sildex Tool
DISTINGUISH THE SYNCHRONOUS LANGUAGES
Distinguish Of The Synchronous Languages

We can distinguish the synchronous languages based on two assumptions:

• Programming Paradigm
• The Philosophy of the languages
Programming Paradigms

Synchronous programming languages

- Imperative language
  - Esterel

- Declarative language
  - Lustre
  - Signal
The philosophies of synchronous language

- Microsteps
  - Very High Speed Integrated Circuit Hardware (VHDL)
  - Verilog modeling languages
  - Harel's Statecharts,
  - Control System (to program programmable logic controllers)
- Acyclic
  - Lustre
- Unique fixpoint
  - Esterel
- Relation or Constraint
  - Signal
Microsteps

We define a reaction as microsteps to confirm it is operational.
Typically applied in:
• Very High Speed Integrated Circuit Hardware (VHDL)
• Verilog modeling languages
• Harel's Statecharts,
• Control System (to program programmable logic controllers)
Acyclic

No directed cycles

We can insist that a system behaves functionally, when the block diagrams of control systems contain no zero-delay loops

- Lustre
Unique fixpoint

Each reaction of a system is assumed to be the solution of a fixpoint equation.

- Esterel
Relation or Constraint

Each reaction of a system is assumed to be a constraint

- Signal
SUCCESES AND IMPROVEMENTS
Successes and Improvements

Lustre

Esterel

Signal
Successes and Improvements : Lustre

- Airbus A320
  - Airbus Industries
- N4 series of nuclear power plants
  - Schneider Electric
Successes and Improvements: Esterel

- Dassault Aviation
  - Landing gear system and a fuel management system
- Simulog
- Texas Instrument
Successes and Improvements: Signal

- CNET
- INRIA
- TNI
- Snecma
Observers for Verification and Testing

• With Observers, we can:
  • Describe non-deterministic behaviors of programs using synchronous observers
    • We use the approach called “an automata-theoretic approach to automatic program verification” (see [11])
      • First, describe the unwanted traces of the program
      • Then, make sure that these traces are unaccepted by the automaton
  • Specify safety properties
  • Observe variables or signals of interest

• Advantages of using observers:
  • We can specify the safety properties within the program itself
  • Observers can be executed (good for testing)
  • It can be run during execution → We can perform auto test
Visual Notations: Examples

Source: http://www.esterel-technologies.com
Visual Notations: Benefits

- Easy to use
- Reusable
- Can be generated into codes
  - E.g., UML (Universal Modeling Language), Simulink/Stateow, and SyncCharts
- Potential features of the languages
DIFFICULTIES AND OPEN ISSUES
Difficulties and Open Issues

- Compilation
  - Esterel
- Handling of arrays
Compilation (Esterel) (1)

- V1, V2, and V3
  - Based on literal interpretation
  - Based on automata using Brzozowski's algorithm
  - Worked good for small a program
  - However, cannot compile concurrent programs that have longer than 1000 lines
- V4
  - Based on automata by translating Esterel into digital logic (can minimize the size of the executable programs)
  - However, incompatible with the prior versions e.g., V3
- V5
  - Slower than automata-based 100 times
- SAXO-RT (Weil et al. [12]) : "compiled-code discrete-event simulators"
  - The program is divided into segments. Each one becomes a separate C function and can be invoked by a centralized scheduler
Compilation (Esterel) (2)

- Certification constraints of the safety-critical software (DO178B)
  - E.g., traceability (mandatory)
- Tradeoff between traceability and efficiency
  - In [1] suggests choose one of them
  - Therefore Scade/Lustre compiler → traceability
Handling Of Arrays

- Powerful to structure programs and to define parameterized regular networks
  - E.g., Apply one operator for the whole element of an array
- First introduced in Lustre to describe circuit (mandatory to manipulate bits)
- An Example Problem (Lustre V4) (See the next slide)
- We have a loop

\[
c = 0;
\text{for} \ (i = 0; \ i <= n; \ i++) \ \{ \\
\quad s[i] = A1(a[i], \ b[i], \ c); \\
\quad c = A2(a[i], \ b[i], \ c);
\}
\]

Source: [1]
An Example Problem (Lustre V4)

Array $a$

Compilation with expansion:

Array elements are expanded into independent variables
SUMMARY
Summary

• The synchronous languages used in the industries as a technology to model, specify, validate, and to implement real-time embedded applications
• We can answer the questions described in the first section
• We have reviewed the history of the synchronous programming languages
  - Esterel, Lustre, Signal
• We have found that languages have been used in many projects:
  • Observers for Verification and Testing
  • Visual Notations
• We have found compilation of Esterel was difficult and still need to improve
• We have found the problem of handling with arrays is still an open issue
• We think the potential features of the languages might be their visual notations
References


References


References


Thank You

Q & A