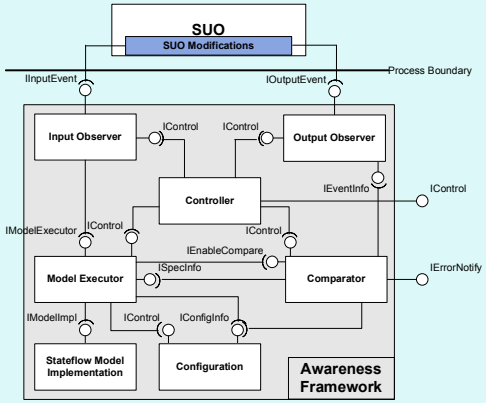
Embedded Systems
INSTITUTE

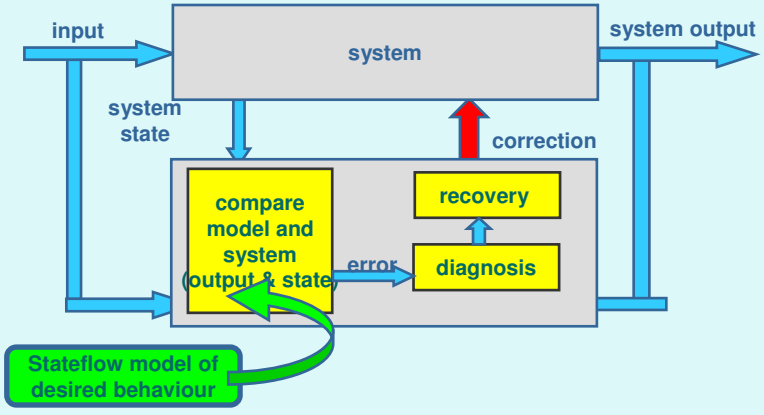
Model-Based Run-Time Error Detection

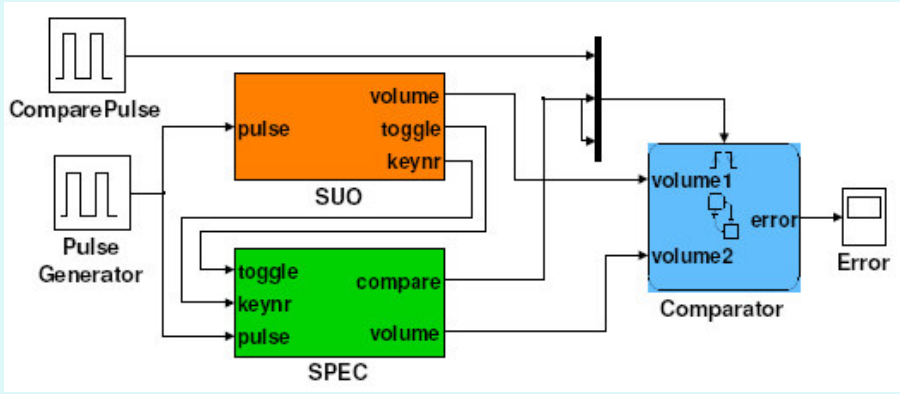
Jozef Hooman & Teun Hendriks

Embedded Systems Institute

Eindhoven, The Netherlands







Models@run.time, 2 October 2007

TRADER
System Reliability

Embedded Systems INSTITUTE

Previously known as Philips Semiconductors

NXP founded by Philips

Carrying Industrial Partner

Period: Sept. 2004 - Aug. 2009
20 fte/yr, 7 PhDs,
1 Postdoc, 10 Partners

TRADER

imec

TU Delft
Delft University of Technology

TU/e technische universiteit eindhoven

DESIGN TECHNOLOGY INSTITUTE

Universiteit Leiden

University of Twente
The Netherlands

Embedded Systems INSTITUTE

TASS software professionals

NXP founded by Philips
Research

PHILIPS
Consumer Electronics

Goal

Develop methods and tools to optimize reliability of high-volume products

Issues

- Minimize product failures.
- Maintain high-level of user-perceived reliability

Case studies from TV domain

Reliability threats in TV domain

Embedded Systems INSTITUTE

Increasing complexity

- **Functions/content increases rapidly**
  - *Play music (mp3, ..), view photos, search teletext, Electronic Programming Guide, child lock, sleep timer, Picture-in-Picture, TV ratings, emergency alerts, many image processing options and user settings, …*
- **External information sources multiply**
  - *Connected planet strategy, downloadable applications*

➔ **Increase of SW (1KB in 1980 – 64MB in 2007)**
Increase of third party content (EPG, codec's)

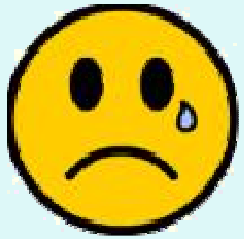
Decreasing time-to-market

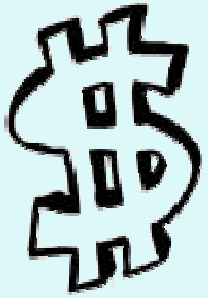
- **Fixed shipping gates to occupy reserved shelf space**

Business impact

Embedded Systems INSTITUTE

- **Not satisfying the high reliability expectations**
  - **Many returned products**
  - **Damages brand image**
  - **Reduces market share**



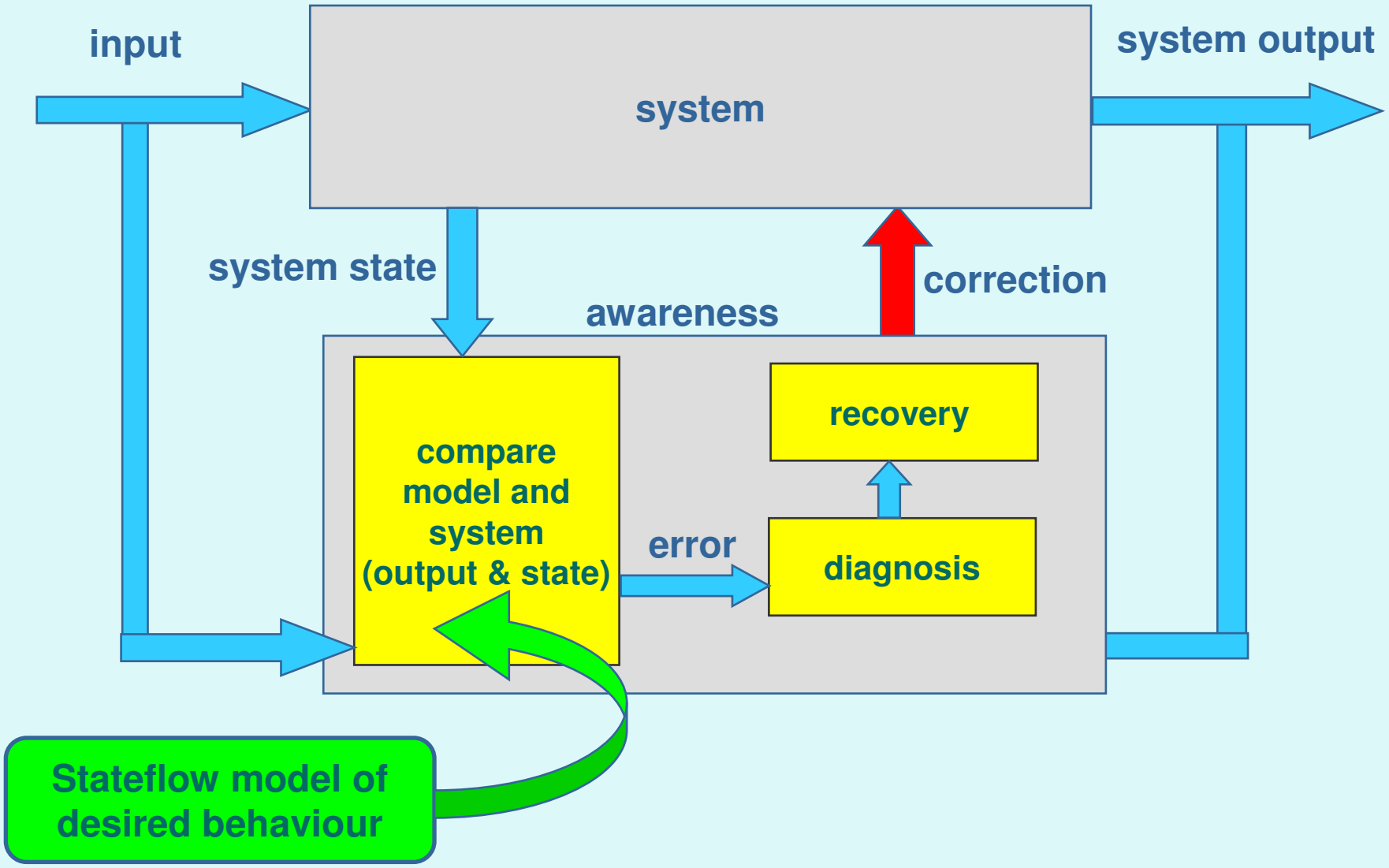


Challenge:

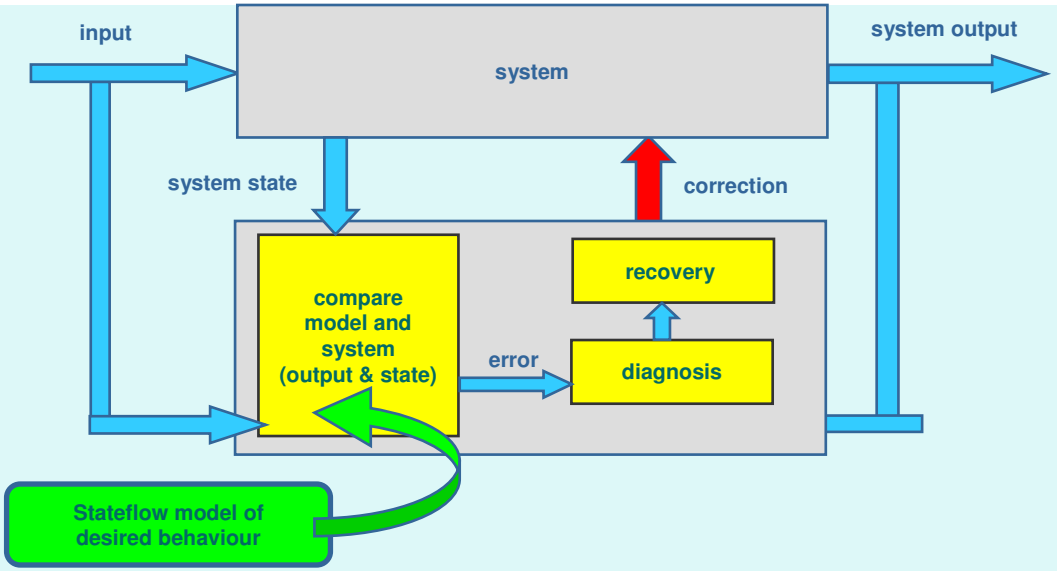
- **Prevent product faults causing customer complaints given constraints:**
  - **Low costs**
  - **Short time to market**

Approach: run-time awareness

Embedded Systems INSTITUTE



Research Questions (1)



Modeling

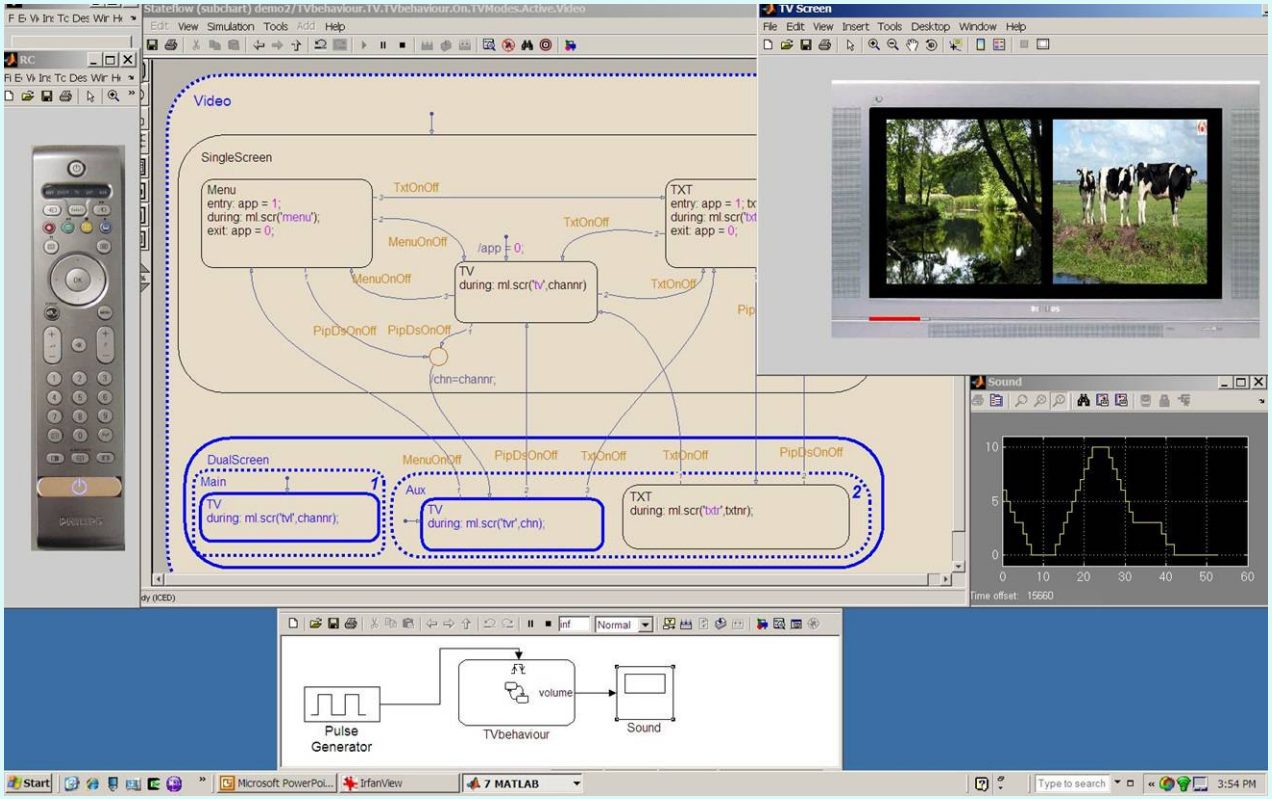
- Which part of the system to model?
- What are most suitable models, at which level of abstraction?
- Which models can be implemented most efficiently?
- How to obtain suitable models?
- How to increase confidence in the model?

Model desired behaviour

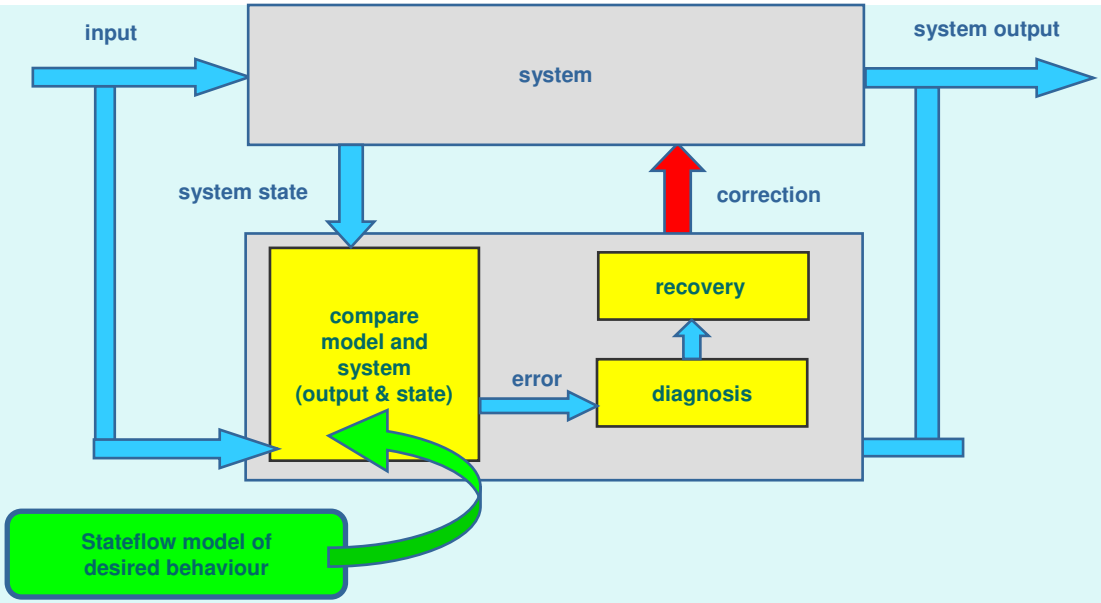
Embedded Systems INSTITUTE

First model desired user-perceived behaviour
(in Matlab/Simulink/Stateflow)

➔ More difficult than expected;
not explicit in current requirements documents



Research Questions (2)



Models at run-time

- How to preserve model semantics in implementation?
- How to avoid false errors?
- When to compare system observations with model?
- When to report an error, how much tolerance is allowed?

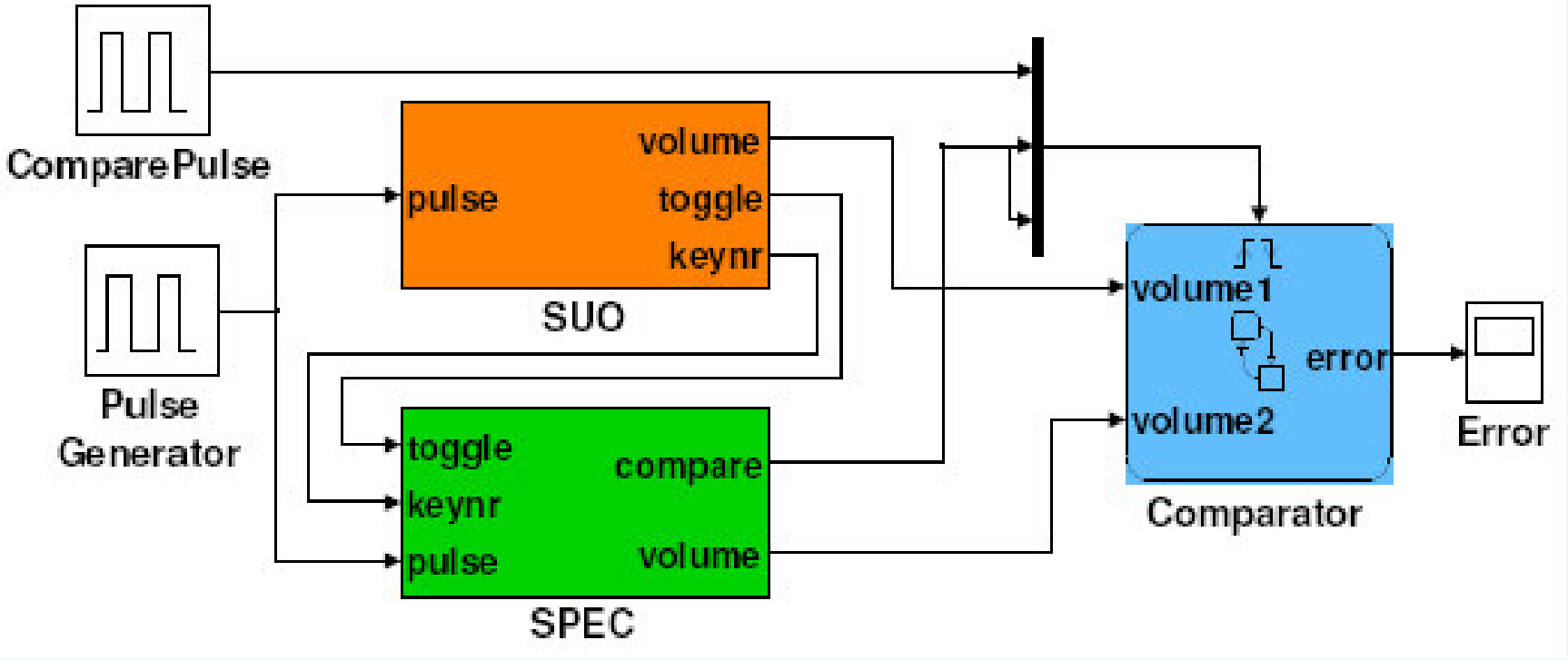
Awareness Experiments

Embedded Systems INSTITUTE

Experiment with awareness concept:

- **Simulation using Simulink/Stateflow**
  compare model of System Under Observation (SUO) and SPEC, try different error detection strategies
- **Linux-based awareness framework**
  in which SUO and SPEC can be inserted easily
- **Open source media player MPlayer [current work]**
  model desired behaviour and insert system and spec in Linux-based framework
- **Add awareness to part of TV system [future work]**

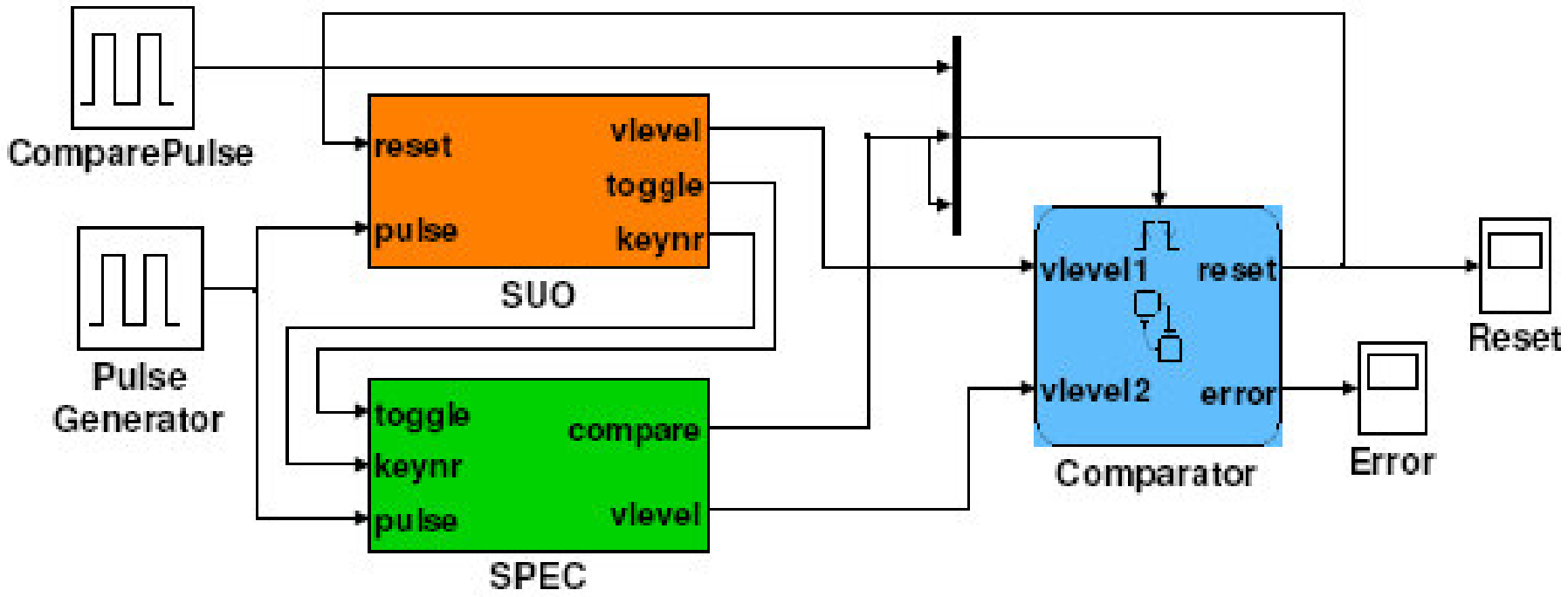
Simulating Awareness



Comparison can be

- Time-triggered (compare pulse)
- Event-triggered (compare signal from model)

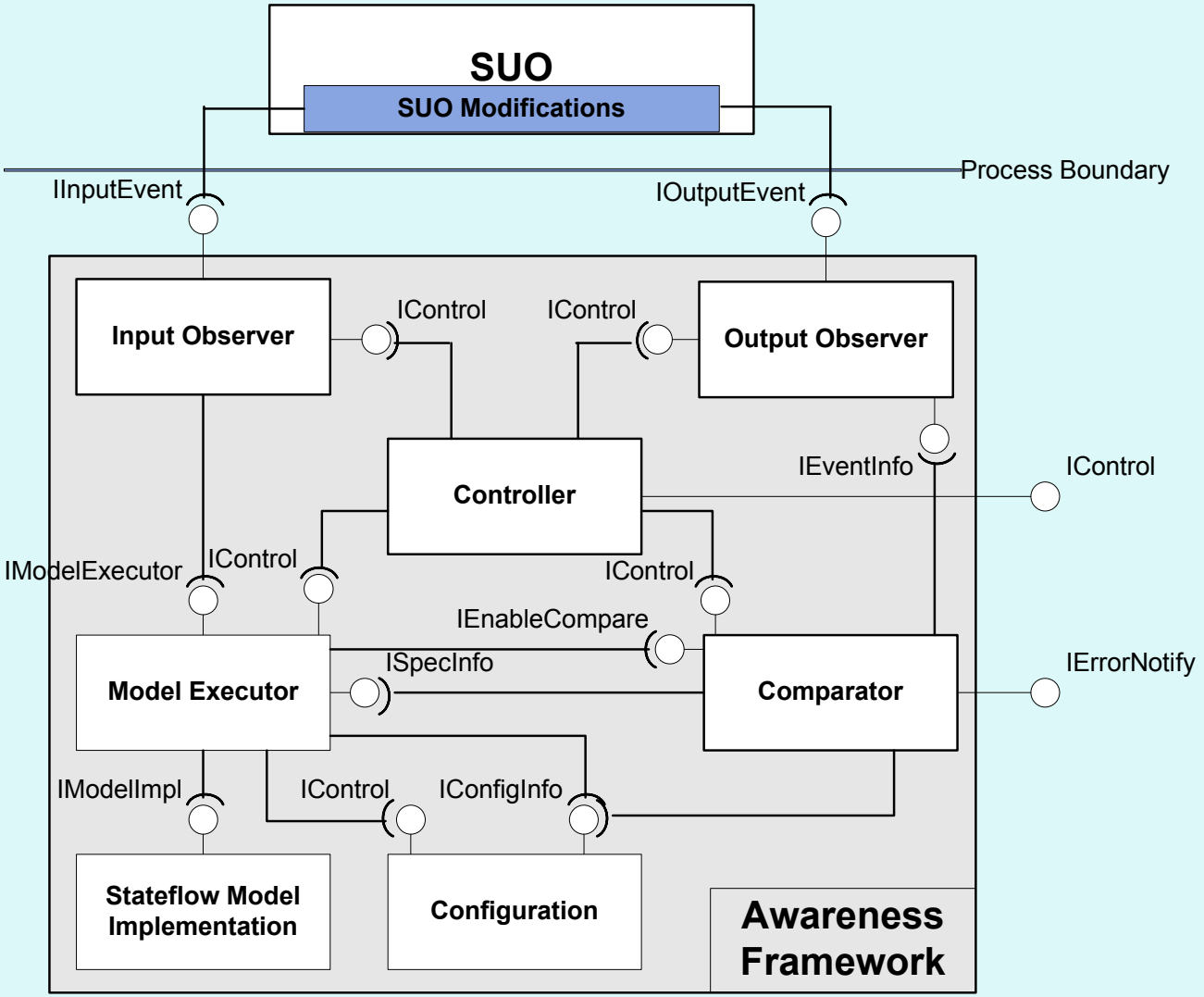
Simulating Awareness



Observe internal variable to detect error earlier and allow recovery before user observes failure

Design of Framework in Linux

Embedded Systems INSTITUTE



Embedded Systems INSTITUTE

Application: MPlayer

Process Boundary

IInputEvent

IOutputEvent

Input Observer

Output Observer

Controller

Model Executor

Comparator

Configuration

Awareness Framework

IControl

IEventInfo

IModelExecutor

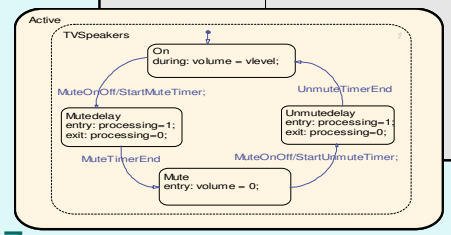
IEnableCompare

ISpecInfo

IErrorNotify

IModelImpl

IConfigInfo



Concluding Remarks

Embedded Systems INSTITUTE

- **Stateflow suitable modeling language**
  - **Allows various forms of model validation**
  - **Various options for code generation**
- **Linux-based framework suitable for experiments**
- **Comparison strategy needs further study**
  - **When to compare to avoid false errors**
  - **In Linux framework we allow the specification of**
    - **Maximal allowed deviation on values**
    - **Maximal number of allowed consecutive erroneous deviations**
  - **Uncertainty about system behaviour, such as worst-case & best-case execution times, are included in model**

    Avoid comparison when systen behaviour is uncertain

Thank you for your attention!

Embedded Systems INSTITUTE