Verification of Quantitative Temporal Properties of SDL Specifications

Iulian Ober, Alain Kerbrat

Telelogic

Telelogic

Overview

- State of the art in timed systems specification/verification
- Timed automata model of SDL: semantics, extensions
- Quantitative temporal property specification
- Timed simulation and verification tool
- Conclusions

Timed systems specification and verification

- *Timed systems =*

  system behavior is triggered by or depends on time

  => time is not only a performance aspect

- Many theoretical models & techniques
  - Behavioral (automata, Petri nets, process algebra, …)

    => model checking
  - Axiomatic (duration calculus, interval calculus, …)

    => rewriting

- Recent work: timed automata – model checking

Formal reasoning about timed systems

Purpose:

- Derive timing estimates
- Prove properties

Prerequisites:

- A formal model for time
- Formal relation between time progress and system execution
- Analysis techniques

Semantics of time in SDL

Z.100 standard

- Loose time progress conditions
  => weak properties ensured
- Complex time-dependent behavior (no restriction on **now**)
  => undecidability

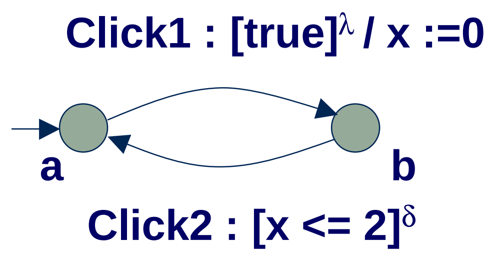
Current simulation and verification tools

- Strong (restrictive) time progress conditions
  => interesting scenarios may not be explored
- Strong restrictions on use of **now** & timers

Telelogic

Timed automata

- TA = FSM + clocks
- Example: *double-click within at most 2 time units*



- Clocks: X = { x,y,… } $\rightarrow$ IR
- Relate execution to time through guards:

  Click2 $\Leftrightarrow$ x $\in$ [0,2]
- Control time progress through *urgency*:

  λ , δ , ε

Telelogic

Timed automata: semantics

- Dynamic states: (q,**v**)
  - discrete: $q \in \mathbf{Q}$
  - time: $\mathbf{v} : \mathbf{X} \rightarrow \mathbf{IR}$
- Transitions:
  - discrete: $(q_1,\mathbf{v}_1) \xrightarrow{\mathbf{e}} (q_2,\mathbf{v}_2)$
  - time: $(q,\mathbf{v}) \xrightarrow{\delta} (q,\mathbf{v} + \delta)$ (i.e. time passes in states)
- Runs – alternation of time & discrete transitions

Telelogic

Timed automata: analysis

- Abstractions of the state space:
  - Region graph, simulation graph, …
- Properties:
  - Reachability, absence of deadlocks, invariance
  - Non-zenoness
  - Model checking timed properties:
    extensions of TL, Timed Büchi Automata
- Decidability limits:
  - More operators
  - Different clock variation laws

Telelogic

SDL and Timed Automata

Mapping

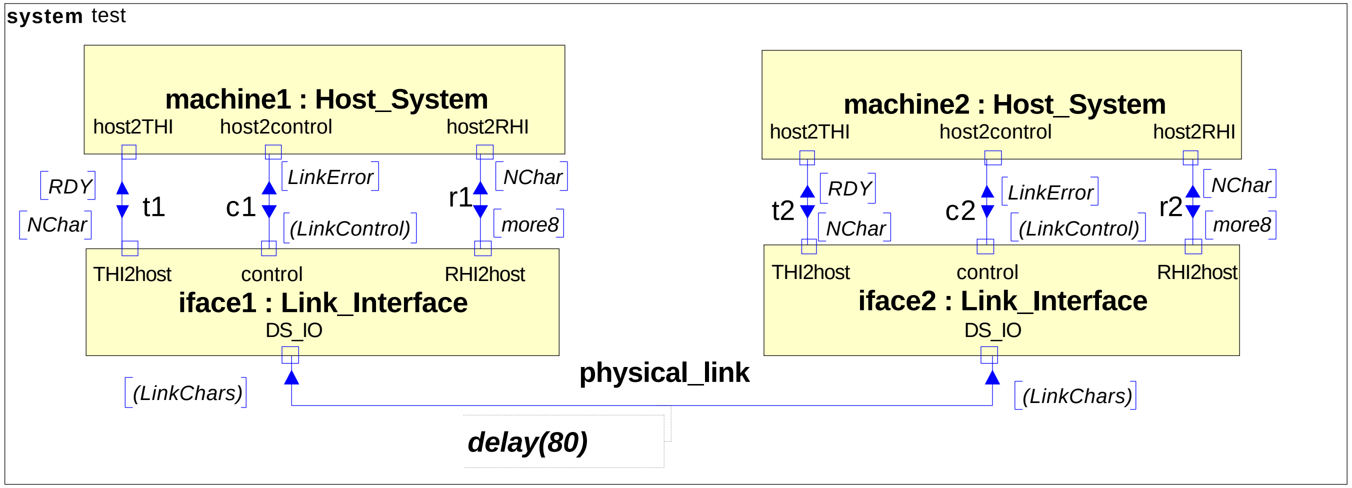
- Timers => clocks
  - set(**now** + d, t) => $x_t := 0$
  - expiration => $[x_t = d]^{\varepsilon}$
- **now** => clock, never reset
- Relative delays measured with **now** => clocks
  - y := **now** => $x_y := 0$
  - (**now** – y) in expressions => x_y

Extensions

- Urgency of transitions, default urgency
- Delaying channels, task execution times

Example: SpaceWire Exchange Level

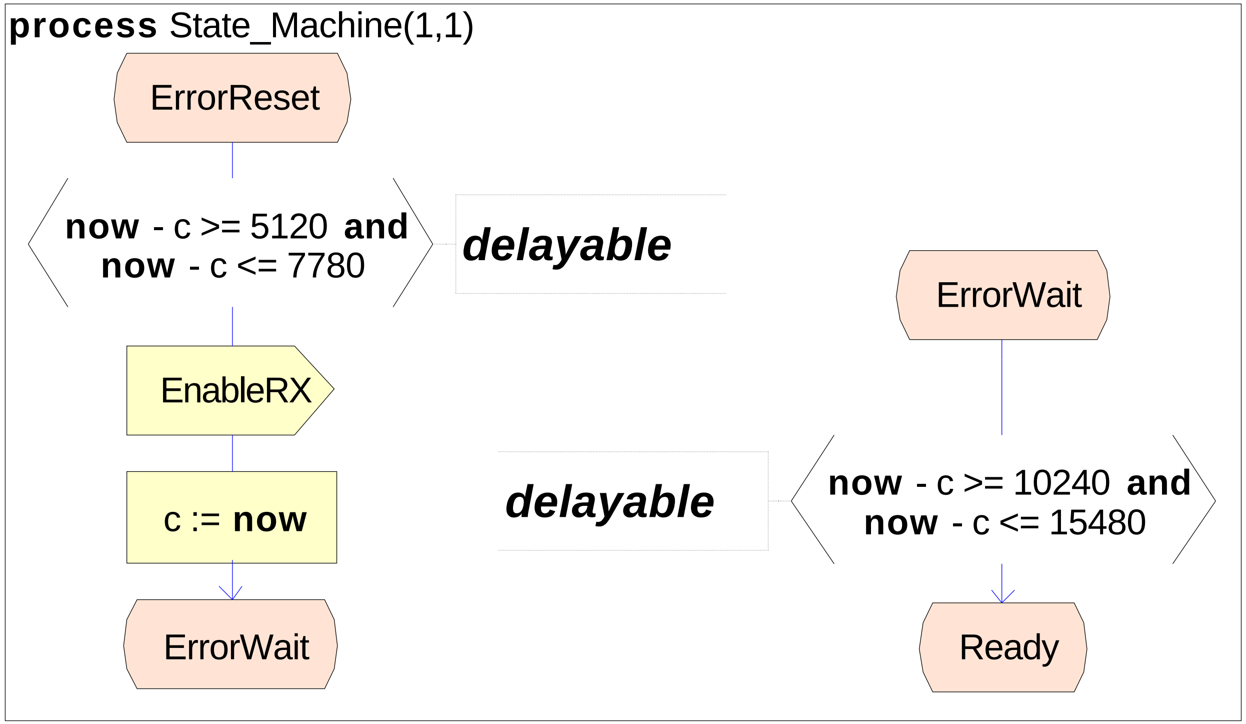
Validation model



Telelogic

Example: delayable urgency

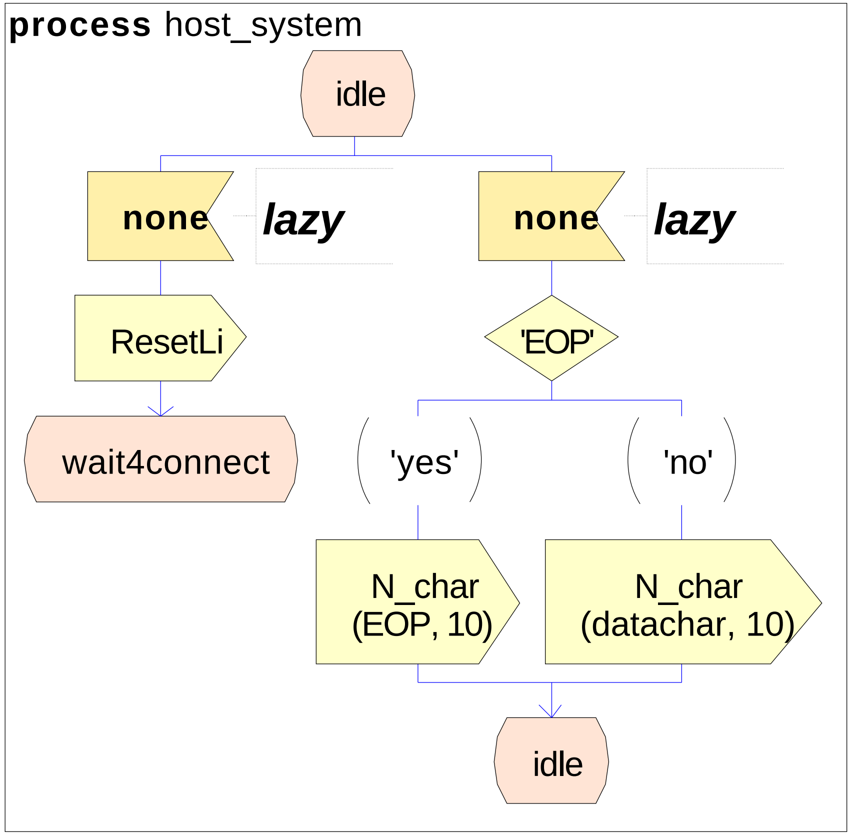
Non-deterministic timing requirements of Link Interface



Telelogic

Example: lazy urgency

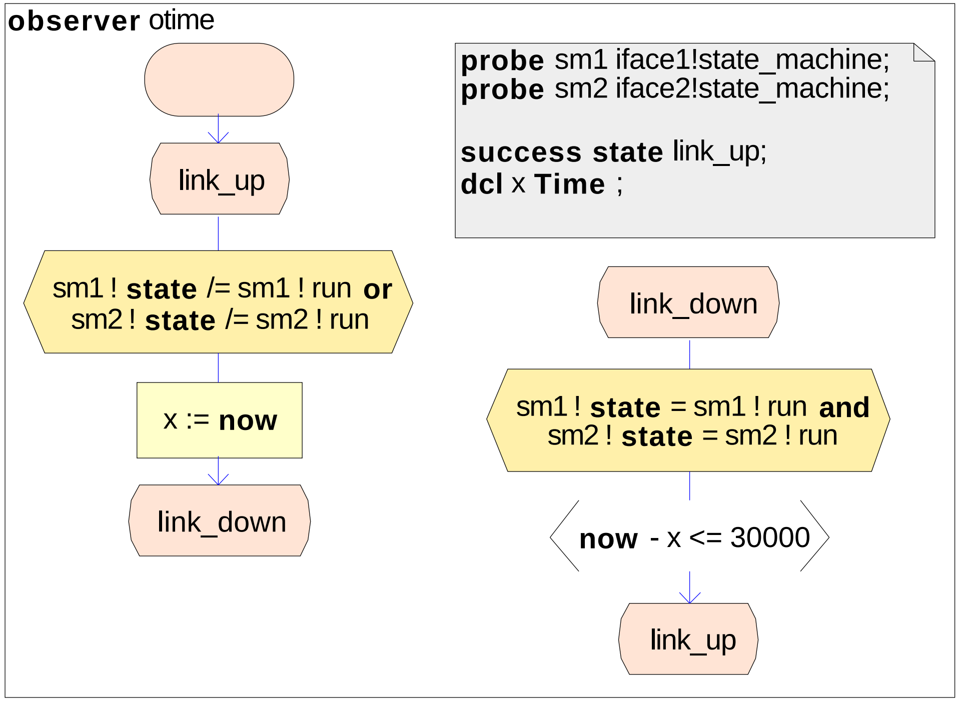
Non-deterministic behavior of the environment



Telelogic

Quantitative temporal properties: GOAL

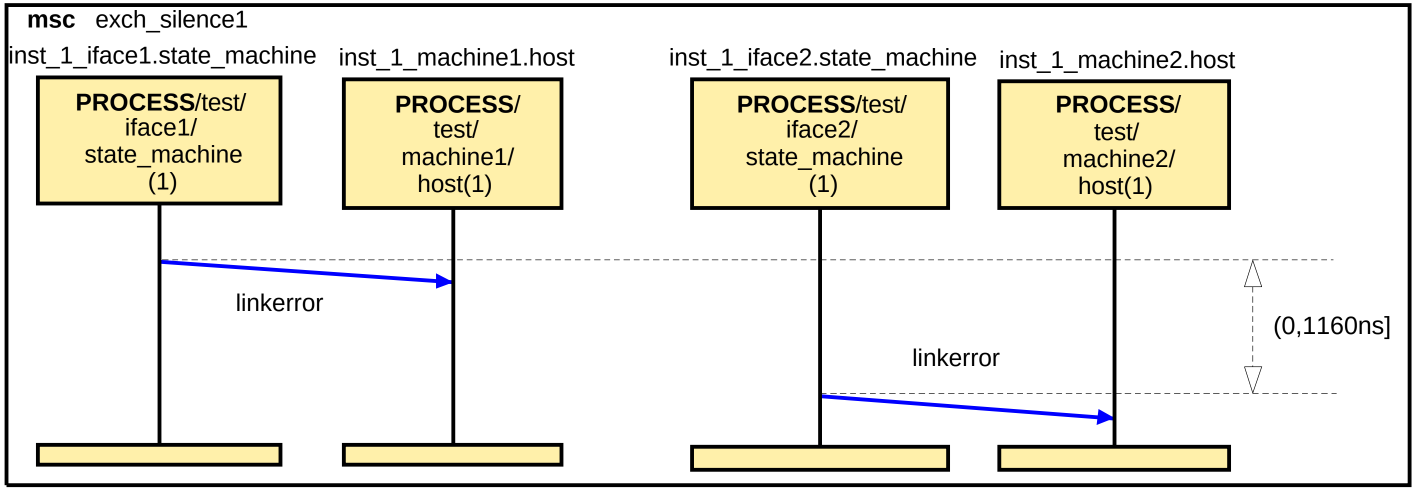
- Simple properties: deadlocks, (timed) invariance (e.g. • (**now** – y <= c))
- Linear properties => GOAL



Telelogic

Quantitative temporal properties: MSC

- Possible use of MSC-2000 time constraints:
  - Define regular subset that may be mapped to TA
  - Define SDL-MSC satisfaction relationship

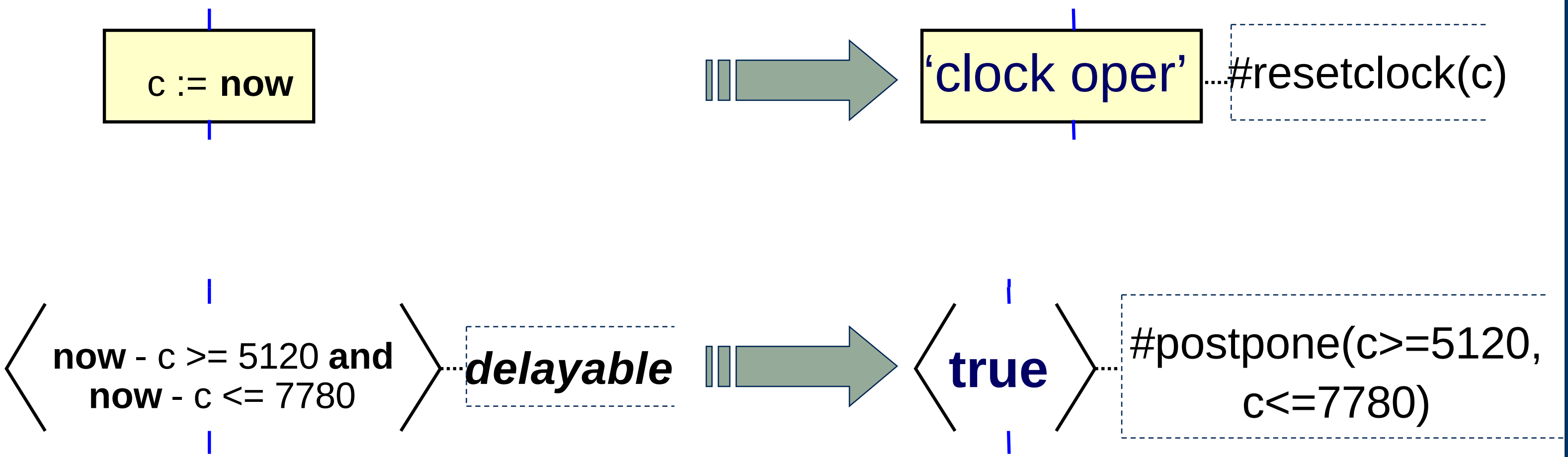


The property verification tool

- Derived from ObjectGEODE simulator

  => implements most of SDL

- Syntactic extensions corresponding to use of now, urgency, etc.



Telelogic

The property verification tool: functioning

- Builds an abstraction of the state space
  => the TA simulation graph
- States: (q, **S**)
  - **S** : the *clock zone* = all the clock configurations reachable in a discrete state => polyhedron in $\mathbb{R}^X$
- Transitions:

$$(q, \mathbf{S}) \xrightarrow{t} (q', \mathbf{S'}) \xrightarrow{\text{time-succ}} (q'', \mathbf{S''})$$

- Synchronous product with GOAL observers
  (built on the fly)

Telelogic

The property verification tool: interface

- Interactive & exhaustive simulation
- Visualization of clock zone:

```
> clocks
0 <= iface1!state_machine!c <= 10440
0 <= iface2!state_machine!c <= 7780
0 <= iface1!state_machine!c - iface2!state_machine!c <= 2660
```

- Can show:
  - time since timers have been set
  - time since a signal has been put in a delaying channel
  - time since a task has begun
  - value of explicit clocks

Telelogic

The property verification tool: interface

- Delaying channel contents:

```
> dchannels
contents of channel physical_link direction towards iface2 =
  1 =
    sender = iface1!transmitter(1)
    name = NChar
    NChar =
      p1 = datachar
```


Telelogic

The property verification tool: interface

- Chronometers for measuring end-to-end delays

```
> addclock chron      -- start interactive measurement
added chronometer chron from console

...                     -- simulation steps

> clocks chron          -- consult chronometer
15360 <= chron <= 23260
> delclock chron        -- remove chronometer
deleted chronometer chron from console
```

Conclusions

- Powerful description and analysis method:
  - Precise control of time progress in simulation
  - Description/simulation of delaying channels, task execution times…
  - Description of timing of the system environment
- Expected gains for user
  - Detecting timing inconsistencies by simulation (and not by testing)
  - Tuning of timers at simulation (not after deployment)
- Weak points:
  - restricted set of operations on **now**
    (e.g. `output s(`**`now`**`)` not supported)
    => adaptive algorithms involving measurements not supported
  - computational complexity of algorithms