KSE 2014

October 10th, 2014
Hanoi

Formalising Concurrent UML State Machines Using Coloured Petri Nets

Étienne André, Mohamed Mahdi Benmoussa, Christine Choppy

Université Paris 13, Sorbonne Paris Cité, LIPN, CNRS, France

Motivation: Complex Systems Safety

- Need for early bug detection
  - Bugs discovered when final testing: expensive

  $\leadsto$ Need for a thorough modelling phase
- Critical and complex systems that need verification
- Specification with UML state machines (SMDs) [OMG, 2011]
- Informal description of UML semantics
- No formal verification on UML state machines
- Solution: Provide a formalisation using coloured Petri nets

Outline

1. Concepts (SMDs & CPNs)
2. Translation of Concurrent State Machines
3. Conclusion and Perspectives

Outline

1. Concepts (SMDs & CPNs)

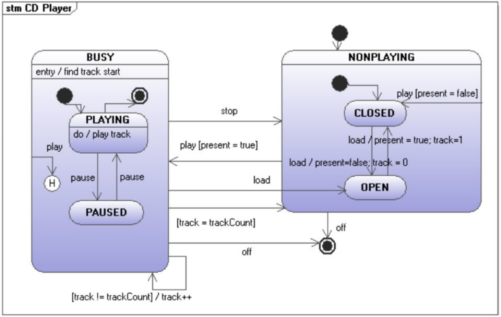
2. Translation of Concurrent State Machines

3. Conclusion and Perspectives

UML Behavioural State Machines

- Transition systems used to express the behaviour of dynamic systems
- Specified in [OMG, 2011]
- Widely used in the industry
- Semantics not formally expressed
  - Informal specification in [OMG, 2011]
  - Not directly suitable for formal methods

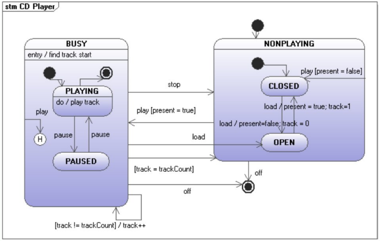
Example of a CD Player [Zhang and Liu, 2010]



- Features
  - A hierarchy of simple and composite states
  - Transitions (including inter-level) with events
  - Entry (`find track start`) and do (`play track`) behaviours
  - Global variables (`present` and `track`)
  - History pseudostate (H)

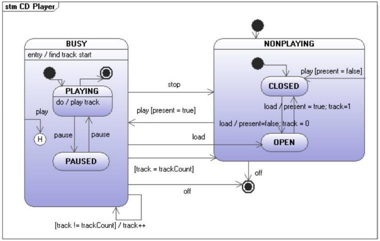
Example of a CD Player (cont.)

- This example is simple
  - Few states, few events, few variables
  - No exit behaviour



Example of a CD Player (cont.)

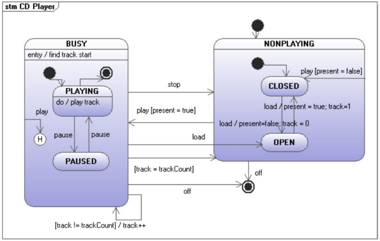
- This example is simple
  - Few states, few events, few variables
  - No exit behaviour



- And still. . . Can we ensure the following?
  - “When in **PLAYING**, there is a CD in the player”
  - “When in **PLAYING**, the track number is always between 1 and trackCount”

Example of a CD Player (cont.)

- This example is simple
  - Few states, few events, few variables
  - No exit behaviour



- And still... Can we ensure the following?
  - "When in **PLAYING**, there is a CD in the player"
  - "When in **PLAYING**, the track number is always between 1 and `trackCount`"

- Not easy to guarantee!
  (So what about larger case studies...)

Main Goal

- We translate UML state machines to coloured Petri nets (CPNs)
- Set of considered constructs
  - Hierarchy of composite states
    - simple
    - orthogonal (with regions)
  - Inter-level transitions
  - Entry, do, exit behaviours with global variables
  - History pseudostates
  - Concurrency (fork, join, synchronization)

Main Goal

- We translate UML state machines to coloured Petri nets (CPNs)
- Set of considered constructs
  - Hierarchy of composite states
    - simple
    - orthogonal (with regions)
  - Inter-level transitions
  - Entry, do, exit behaviours with global variables
  - History pseudostates
  - Concurrency (fork, join, synchronization)

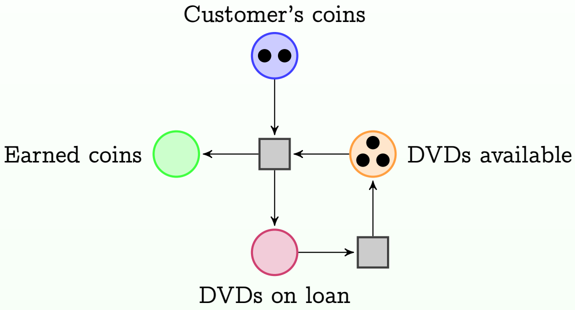
Petri Nets [Petri, 1962]

- A kind of automaton
  - Bipartite graph with places and transitions
  - Tokens can be added to places
    - Represent data or control
  - A state (configuration) of the Petri net: a marking
    - Number of tokens in each place
    - Evolves when firing transitions
  - Initial state: initial marking

- Advantages of Petri nets
  - Detailed view of the process with an expressive graphical representation
  - A formal semantics
  - Powerful tools to simulate and verify the model w.r.t. various properties (reachability, boundedness, invariants, deadlock-freeness, etc.)

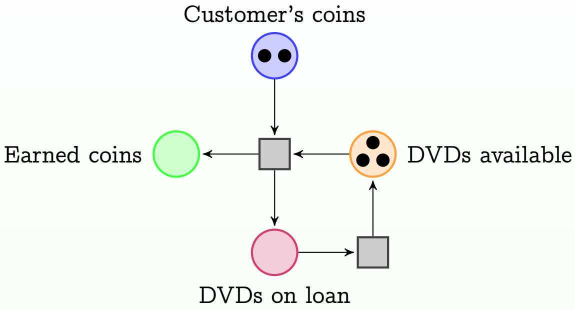
Petri Nets: An Example

A DVD renting machine



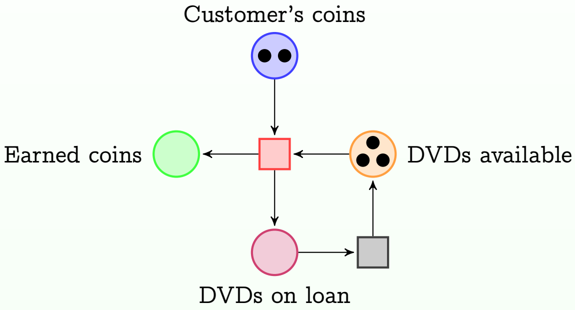
Petri Nets: An Example

A DVD renting machine



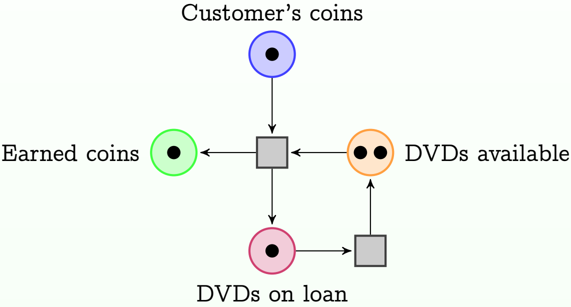
Petri Nets: An Example

A DVD renting machine



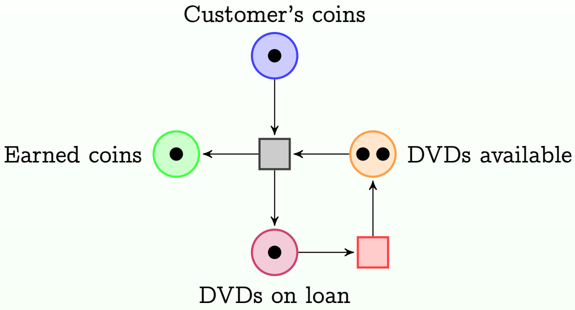
Petri Nets: An Example

A DVD renting machine



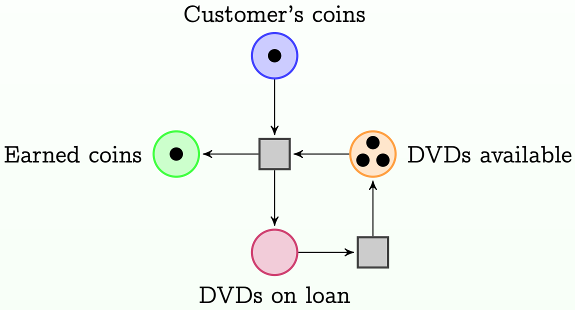
Petri Nets: An Example

A DVD renting machine



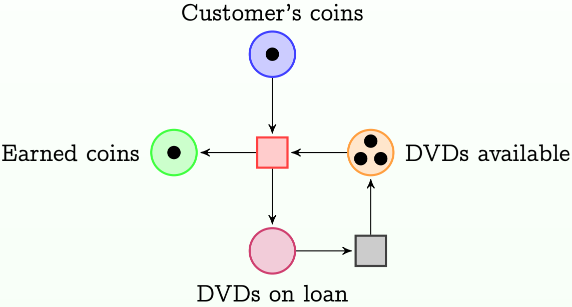
Petri Nets: An Example

A DVD renting machine



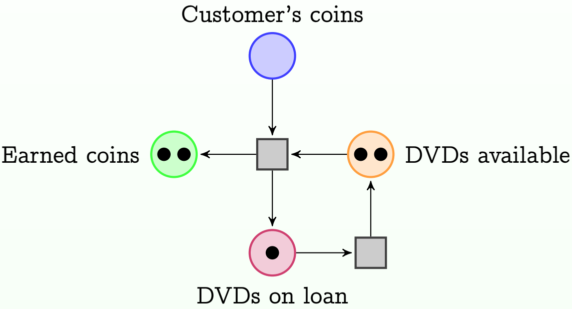
Petri Nets: An Example

A DVD renting machine



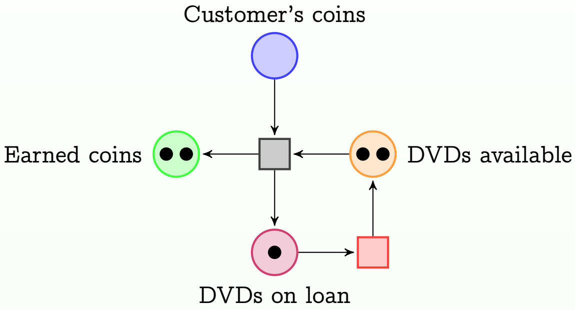
Petri Nets: An Example

A DVD renting machine



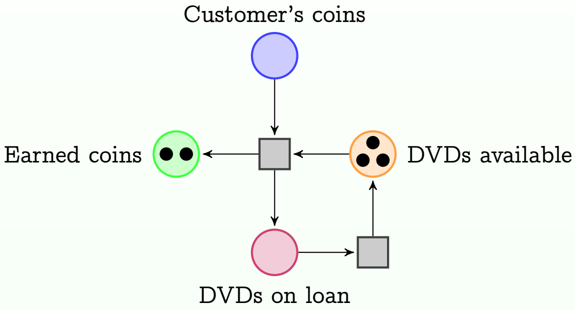
Petri Nets: An Example

A DVD renting machine



Petri Nets: An Example

A DVD renting machine



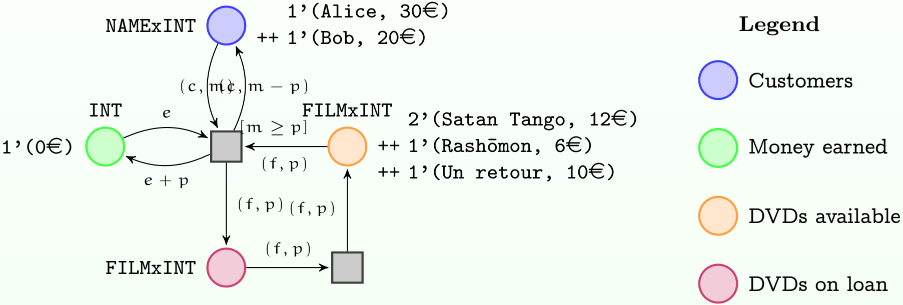
Coloured Petri Nets [Jensen and Kristensen, 2009]

- Extension of Petri nets with colours
  - Tokens and places have a type (“colour set”)
  - Arcs are labelled with expressions
  - Transitions can have a guard

Coloured Petri Nets [Jensen and Kristensen, 2009]

- Extension of Petri nets with colours
  - Tokens and places have a type (“colour set”)
  - Arcs are labelled with expressions
  - Transitions can have a guard

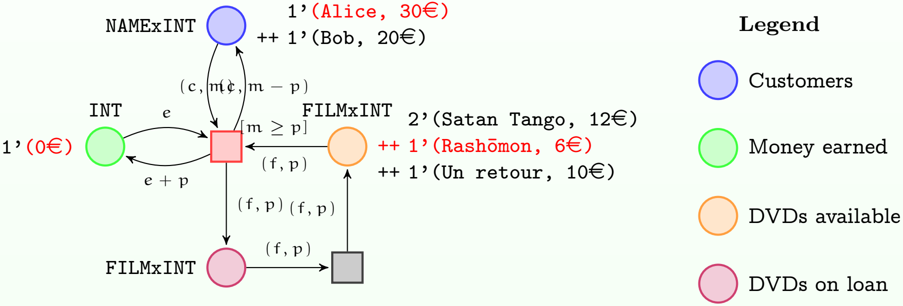
- Example: A more complex version of the DVD renting machine



Coloured Petri Nets [Jensen and Kristensen, 2009]

- Extension of Petri nets with colours
  - Tokens and places have a type (“colour set”)
  - Arcs are labelled with expressions
  - Transitions can have a guard

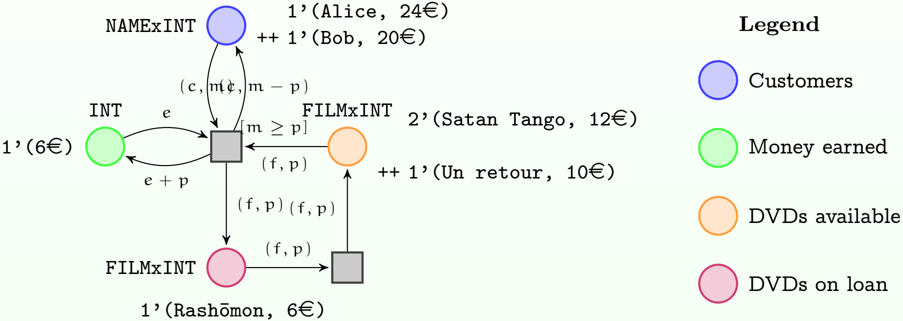
- Example: A more complex version of the DVD renting machine



Coloured Petri Nets [Jensen and Kristensen, 2009]

- Extension of Petri nets with colours
  - Tokens and places have a type (“colour set”)
  - Arcs are labelled with expressions
  - Transitions can have a guard

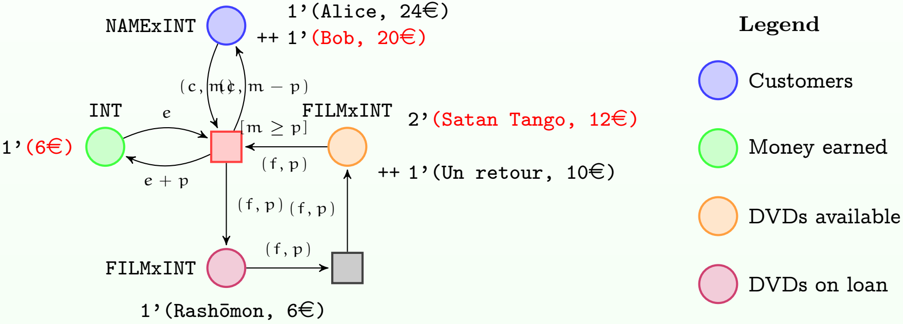
- Example: A more complex version of the DVD renting machine



Coloured Petri Nets [Jensen and Kristensen, 2009]

- Extension of Petri nets with colours
  - Tokens and places have a type (“colour set”)
  - Arcs are labelled with expressions
  - Transitions can have a guard

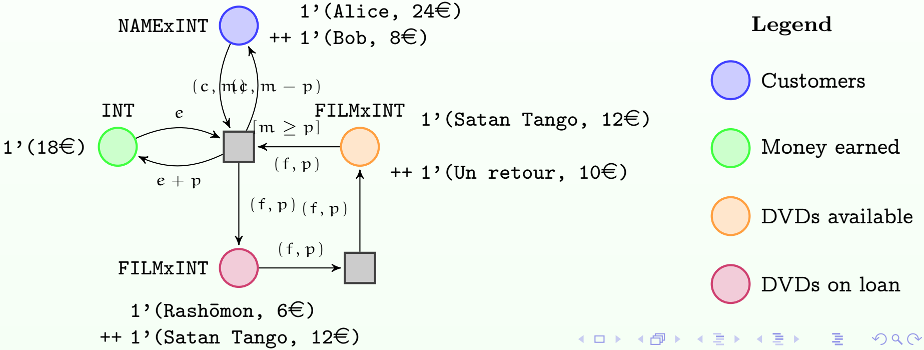
- Example: A more complex version of the DVD renting machine



Coloured Petri Nets [Jensen and Kristensen, 2009]

- Extension of Petri nets with colours
  - Tokens and places have a type ("colour set")
  - Arcs are labelled with expressions
  - Transitions can have a guard

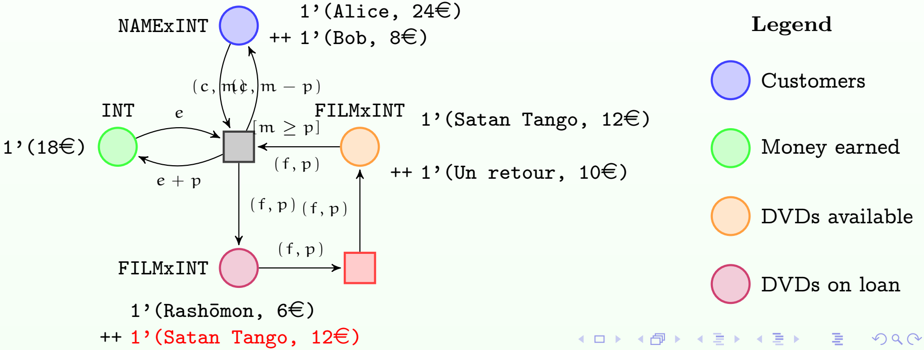
- Example: A more complex version of the DVD renting machine



Coloured Petri Nets [Jensen and Kristensen, 2009]

- Extension of Petri nets with colours
  - Tokens and places have a type (“colour set”)
  - Arcs are labelled with expressions
  - Transitions can have a guard

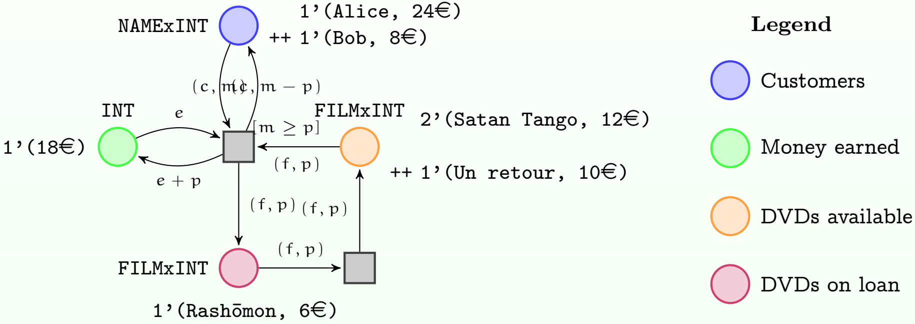
- Example: A more complex version of the DVD renting machine



Coloured Petri Nets [Jensen and Kristensen, 2009]

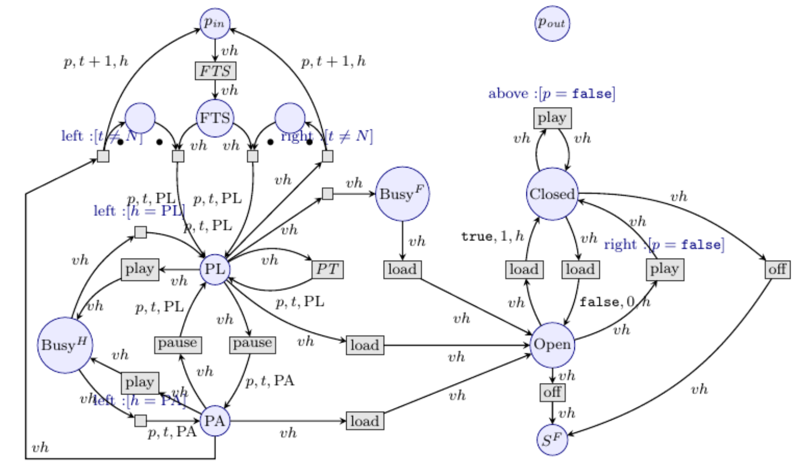
- Extension of Petri nets with colours
  - Tokens and places have a type (“colour set”)
  - Arcs are labelled with expressions
  - Transitions can have a guard

- Example: A more complex version of the DVD renting machine



An Example of a CPN

(Partial) translation of the CD player

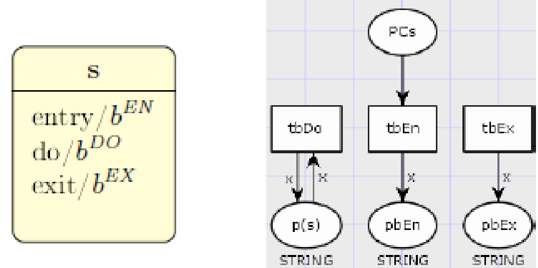


General translation Scheme

- Deeply revise & extend translation from SMDs to CPNs proposed in [André, Choppy, Klai, 2012]
- Tricky: ways to get a reasonable size for the CPN
- To allow to take into account concurrency, fork and join pseudo-states and orthogonal regions.
- Translation based on three algorithms :
  - Algorithm 1 for states and behaviours
  - Algorithm 2 for transitions
  - Algorithm 3 for history pseudo-states

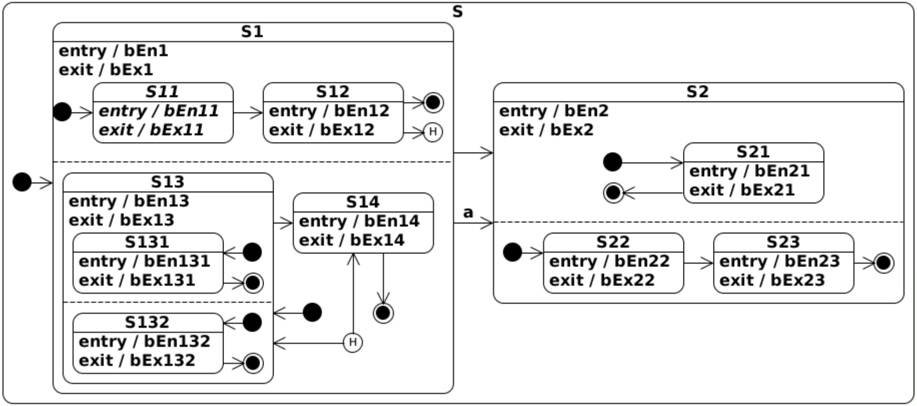
Algorithm 1 -Translation of States and Behaviours

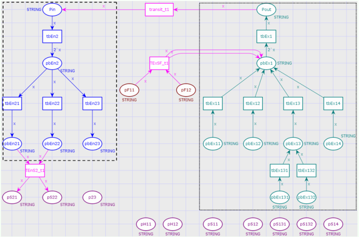
- Each simple, final, history (pseudo) state → one place
- Each behaviour (entry/exit/do) → place, transition & arc
- Hierarchical structure → tree of exit/entry behaviours (used later to connect transitions)



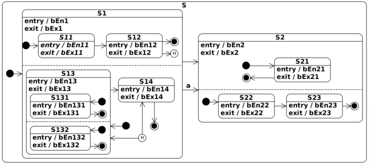
Algorithm 2 - Translation of Transitions

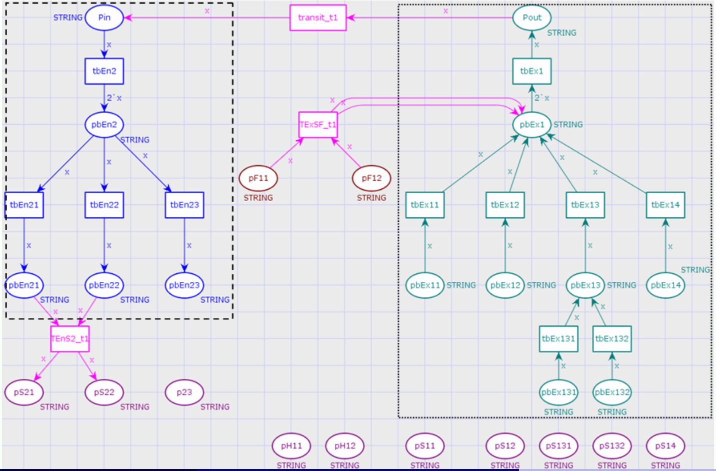
- Establishes connection between source and target of the transition
- Processing differs for simple or composite states (as source and/or target)
- Each UML transition is represented by a CPN transition
- Special processing for concurrent constructs
- Use guards mechanism to guide the tokens in the CPN


S
S1
entry / bEn1
exit / bEx1
S11
entry / bEn11
exit / bEx11
S12
entry / bEn12
exit / bEx12
H
S13
entry / bEn13
exit / bEx13
S131
entry / bEn131
exit / bEx131
S132
entry / bEn132
exit / bEx132
S14
entry / bEn14
exit / bEx14
H
a
S2
entry / bEn2
exit / bEx2
S21
entry / bEn21
exit / bEx21
S22
entry / bEn22
exit / bEx22
S23
entry / bEn23
exit / bEx23


Pin
STRING
x
tbEn2
2`x
pbEn2
tbEn21
tbEn22
tbEn23
pbEn21
pbEn22
pbEn23
TEnS2_t1
pS21
pS22
p23
transit_t1
TExSF_t1
pF11
pF12
Pout
tbEx1
pbEx1
tbEx11
tbEx12
tbEx13
tbEx14
pbEx11
pbEx12
pbEx13
pbEx14
tbEx131
tbEx132
pbEx131
pbEx132
pH11
pH12
pS11
pS12
pS131
pS132
pS14

Example (followed)





Algorithm 3 - History pseudo-states

- Take into account shallow pseudo-states
- Each history is represented by a global variable that contains the last state visited in the region and place/transition
- Variable value updated for each transition fired within a composite state
- Achieved with Algorithm 3 (not represented in the paper for lack of space)

Outline

1 Concepts (SMDs & CPNs)

2 Translation of Concurrent State Machines

3 Conclusion and Perspectives

Element	Considered?
Simple / composite states	Yes
Orthogonal regions	Yes
Initial / final (pseudo)states	Yes
Terminate pseudostate	No (but trivially extensible)
Shallow history states	Identical to [André et al., 2012]
Deep history states	No (but trivially extensible)
Submachine states	No (but trivially extensible)
Entry / exit points	No (but probably easy)
Entry / exit / do behaviours	Yes
Shared variables	Yes
External / local / internal transitions	Yes
Basic fork / joins	Yes
Implicit fork/joins	No (but trivially extensible)
Choices / merges	No (but probably easy)
Deferred events	No
Timing aspects	No

Conclusion and Perspectives

- Extended set of syntactic elements (including hierarchy & concurrency) taken into account
- Developing a tool for implementation of the translation
    - Translation using Acceleo (model to text tool) to implement the translation (done)
    - Home-made tool using Java with parsers to manage the input and the output (partially done)
- Integration of timed events
- Take into account real time specifications
- Prove the formal equivalence between an SMD and its CPN translation

Bibliography

References I

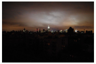
- André, É., Choppy, C., and Klai, K. (2012).
  Formalizing non-concurrent UML state machines using colored Petri nets.
  ACM SIGSOFT Software Engineering Notes, 37(4):1–8.
  UML&FM 2012.
- Jensen, K. and Kristensen, L. M. (2009).
  Coloured Petri Nets – Modelling and Validation of Concurrent Systems.
  Springer.
- Liu, S., Liu, Y., André, É., Choppy, C., Sun, J., Wadhwa, B., and Dong, J. S. (2013).
  A formal semantics for the complete syntax of UML state machines with communications.
  In *iFM*, volume 7940 of *Lecture Notes in Computer Science*, pages 331–346. Springer.
- OMG (2011).
  OMG unified modeling language (OMG UML) superstructure. version 2.4.1, 2011-08-06.
- Petri, C. A. (1962).
  Kommunikation mit Automaten.
  PhD thesis, Darmstadt University of Technology, Germany.

References II

Zhang, S. and Liu, Y. (2010).
An automatic approach to model checking UML state machines.
In *SSIRI (Companion)*, pages 1–6. IEEE Computer Society.

Additional explanation

Explanation for the 4 pictures in the beginning



Allusion to the Northeast blackout (USA, 2003)
Computer bug
Consequences: 11 fatalities, huge cost
(Picture actually from the Sandy Hurricane, 2012)

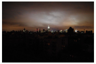
Allusion to any plane crash
(Picture actually from the happy-ending US Airways Flight 1549, 2009)

Allusion to the sinking of the Sleipner A offshore platform (Norway, 1991)
No fatalities
Computer bug: inaccurate finite element analysis modeling
(Picture actually from the Deepwater Horizon Offshore Drilling Platform)

Allusion to the MIM-104 Patriot Missile Failure (Iraq, 1991)
28 fatalities, hundreds of injured
Computer bug: software error (clock drift)
(Picture of an actual MIM-104 Patriot Missile, though not the one of 1991)

Licensing

Source of the graphics used



Title: Hurricane Sandy Blackout New York Skyline
Author: David Shankbone
Source: https://commons.wikimedia.org/wiki/File:Hurricane_Sandy_Blackout_New_York_Skyline.JPG
License: CC BY 3.0

Title: Miracle on the Hudson
Author: Janis Krums (cropped by Étienne André)
Source: https://secure.flickr.com/photos/davidwatts1978/3199405401/
License: CC BY 2.0

Title: Deepwater Horizon Offshore Drilling Platform on Fire
Author: ideum
Source: https://secure.flickr.com/photos/ideum/4711481781/
License: CC BY-SA 2.0

Title: DA-SC-88-01663
Author: imcomkorea
Source: https://secure.flickr.com/photos/imcomkorea/3017886760/
License: CC BY-NC-ND 2.0

License of this document

(LaTeX source available on demand)

Authors: Étienne André and Mohamed Mahdi Benmoussa