ESSS 2014

May 13th, 2014
Singapore

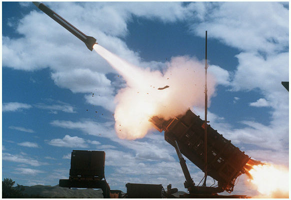
Translating UML State Machines to Coloured Petri Nets Using Acceleo: A Report

Étienne André, Mohamed Mahdi Benmoussa, Christine Choppy

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

Context: Complex Systems Safety (1/2)

- Need for early bug detection
  - Bugs discovered when final testing: expensive
  - ⇝ Need for a thorough modelling phase



Context: Complex Systems Safety (2/2)

- Critical and complex systems that need verification
- Specification with UML state machines (SMDs) [OMG, 2011]
- Informal description of UML semantics
- Solution: Model translation to another formalism

Outline

1. Concepts
2. Towards Model Transformation
3. Translation Using Acceleo
4. Conclusion and Perspectives

Outline

1 Concepts

2 Towards Model Transformation

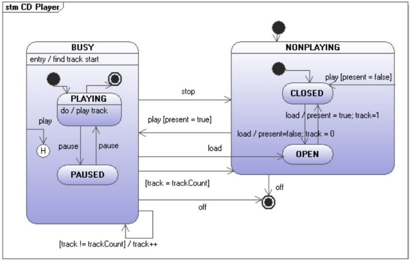
3 Translation Using Acceleo

4 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

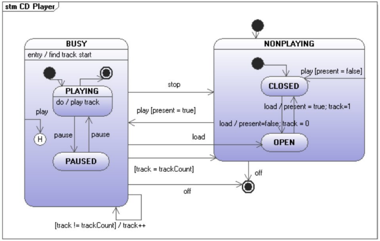
Example of a CD Player [Zhang and Liu, 2010]



- 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`)

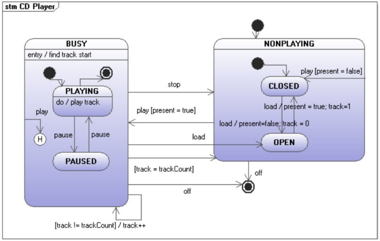
Example of a CD Player (cont.)

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



Example of a CD Player (cont.)

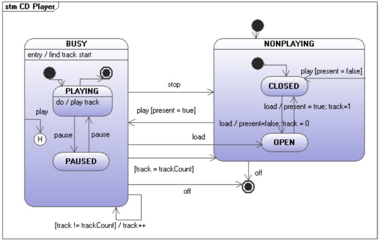
- This example is simple
    - Few states, few events, few variables
    - No concurrency
    - No exit behaviour



- 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 concurrency
  - 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 choose here to use the translation of UML state machines to coloured Petri nets (CPNs) [A., Choppy, Klai, 2012]
- Set of considered constructs
  - Hierarchy of composite states
  - Inter-level transitions
  - Entry, do, exit behaviours with global variables
  - History pseudostates
  - No concurrency (no fork, join, synchronization)

Main Goal

- We choose here to use the translation of UML state machines to coloured Petri nets (CPNs) [A., Choppy, Klai, 2012]
- Set of considered constructs
  - Hierarchy of composite states
  - Inter-level transitions
  - Entry, do, exit behaviours with global variables
  - History pseudostates
  - No concurrency (no fork, join, synchronization)

Goal

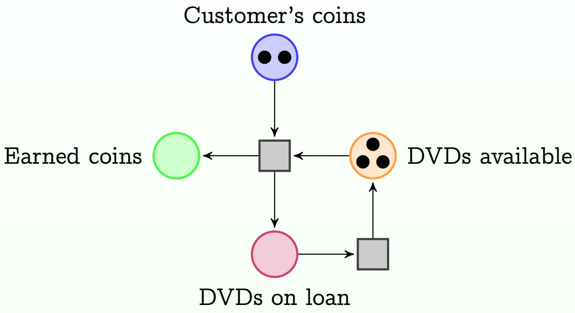
"Implement the translation of [A., Choppy, Klai, 2012] ."

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.)

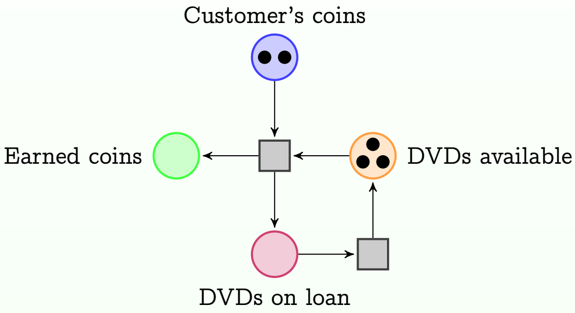
Petri Nets: An Example

A DVD renting machine



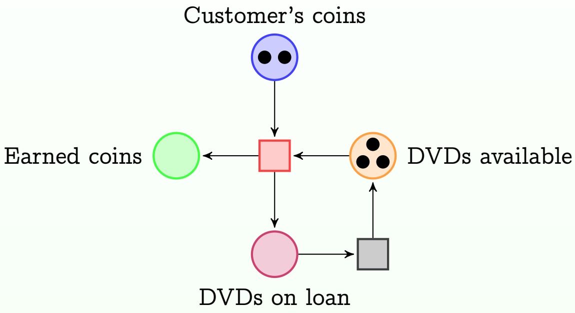
Petri Nets: An Example

A DVD renting machine



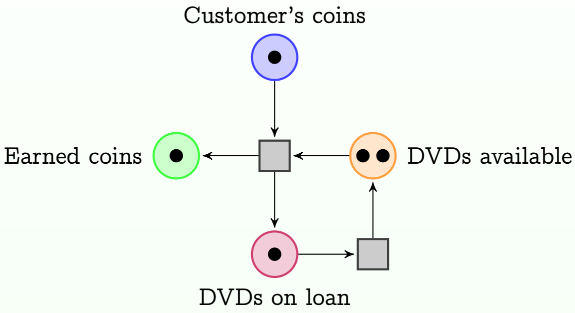
Petri Nets: An Example

A DVD renting machine



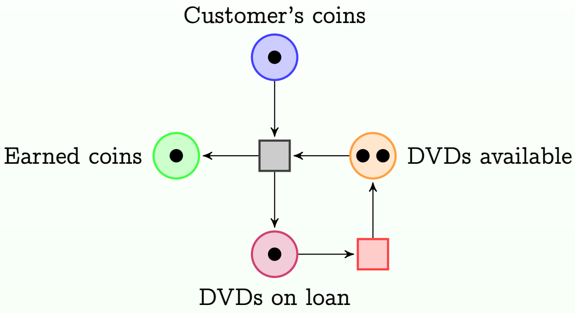
Petri Nets: An Example

A DVD renting machine



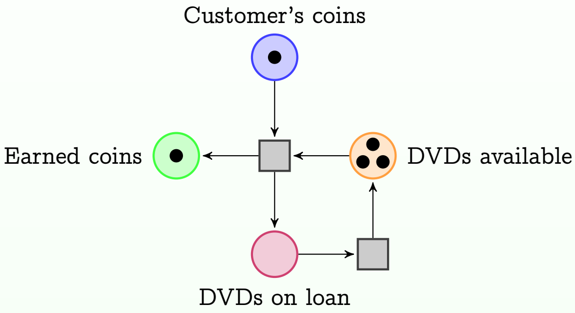
Petri Nets: An Example

A DVD renting machine



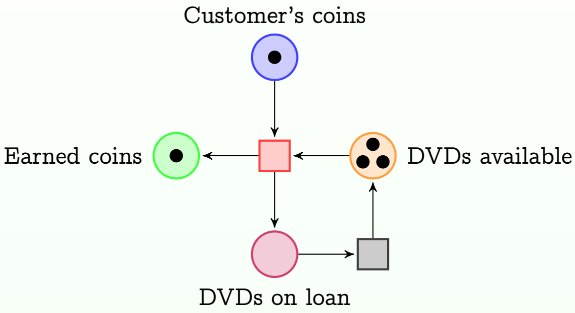
Petri Nets: An Example

A DVD renting machine



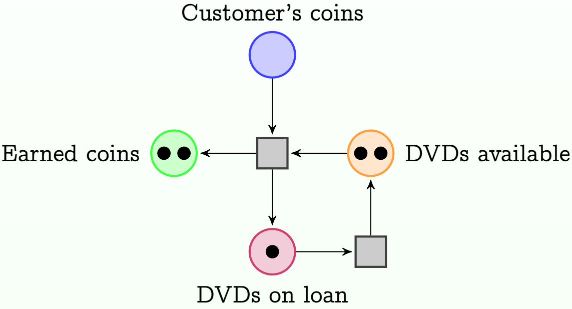
Petri Nets: An Example

A DVD renting machine



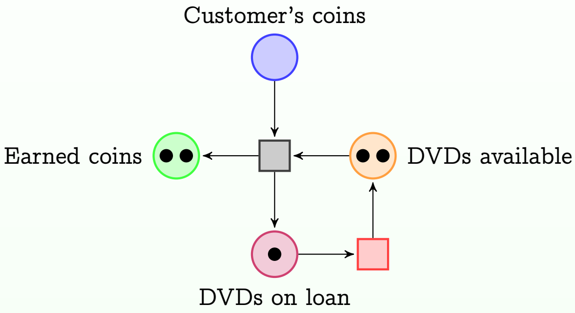
Petri Nets: An Example

A DVD renting machine



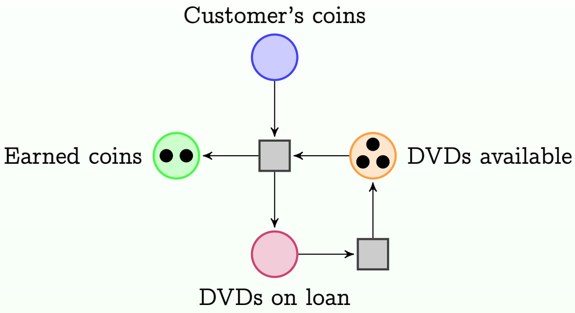
Petri Nets: An Example

A DVD renting machine



Petri Nets: An Example

A DVD renting machine



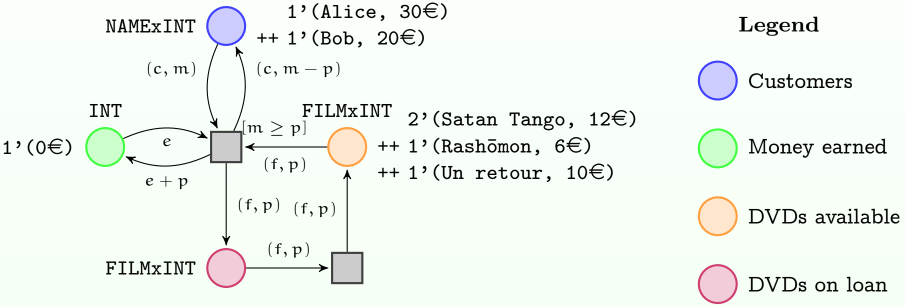
Coloured Petri Nets [Jensen and Kristensen, 2009]

- 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

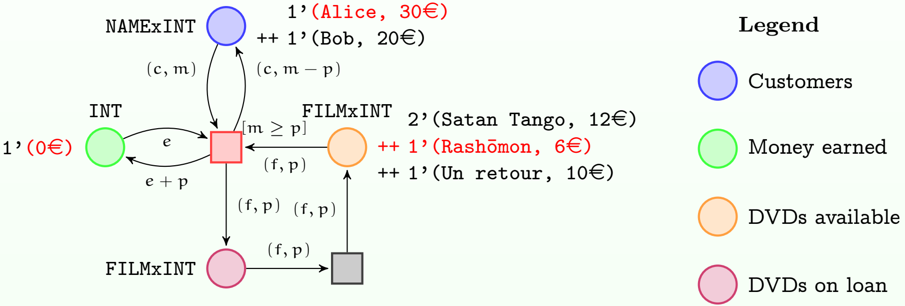
- Example: A more complex version of the DVD renting machine



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

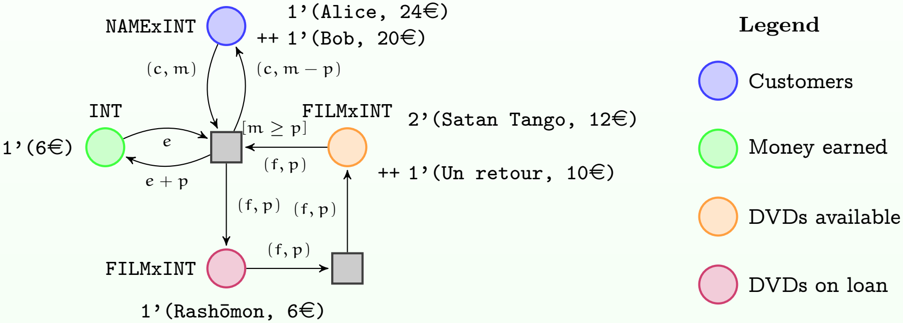
- Example: A more complex version of the DVD renting machine



Coloured Petri Nets [Jensen and Kristensen, 2009]

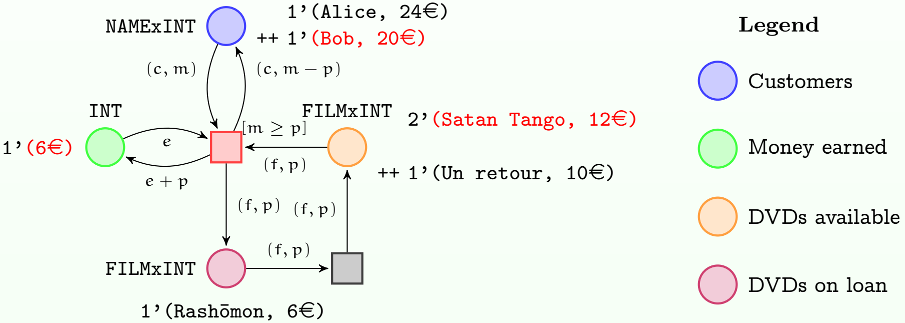
- Extension of Petri nets with colours
  - Tokens and places have a type (“colour set”)
  - Arcs are labelled with expressions
  - Transitions can have a guard

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



Coloured Petri Nets [Jensen and Kristensen, 2009]

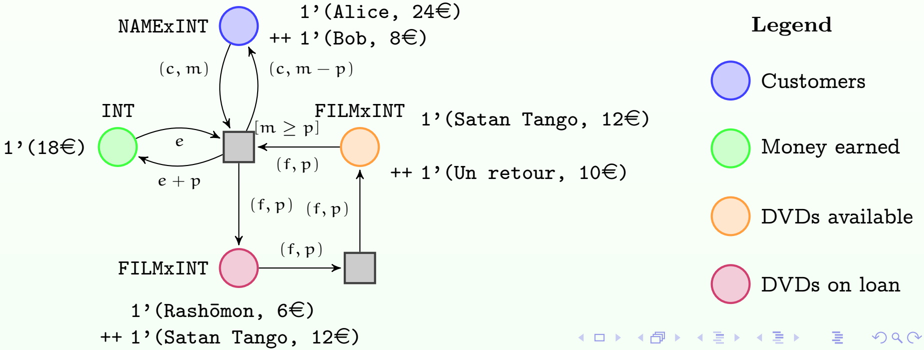
- Extension of Petri nets with colours
  - Tokens and places have a type ("colour set")
  - Arcs are labelled with expressions
  - Transitions can have a guard
- Example: A more complex version of the DVD renting machine



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

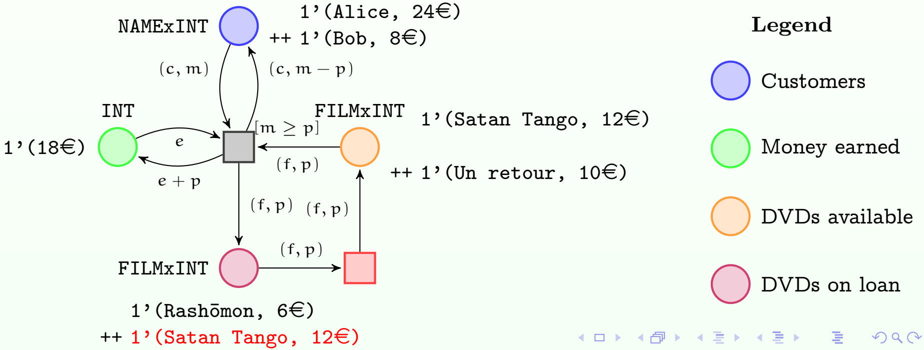
- Example: A more complex version of the DVD renting machine



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

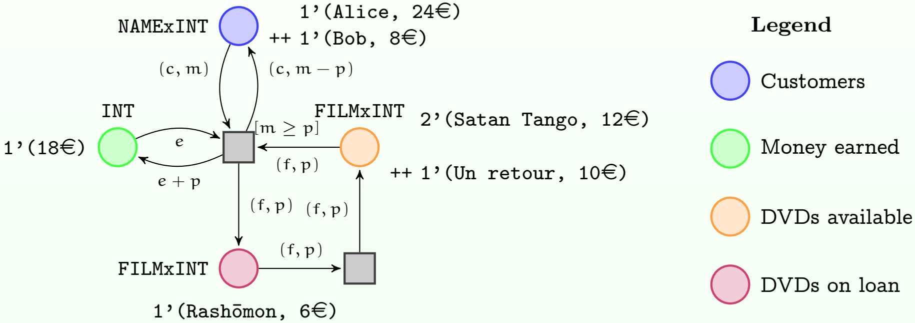
- Example: A more complex version of the DVD renting machine



Coloured Petri Nets [Jensen and Kristensen, 2009]

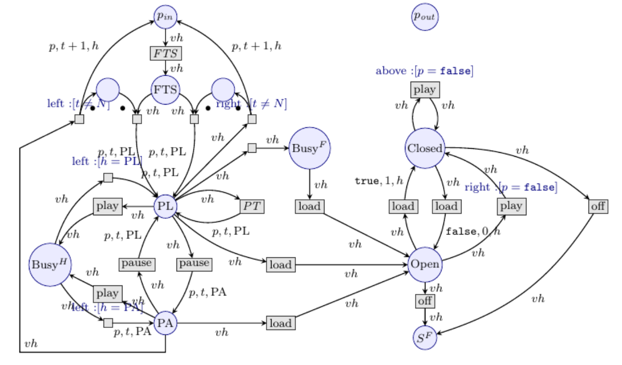
- Extension of Petri nets with colours
  - Tokens and places have a type ("colour set")
  - Arcs are labelled with expressions
  - Transitions can have a guard

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



An Example of a CPN

(Partial) translation of the CD player according to [A., Choppy, Klai, 2012]



Outline

1. Concepts
2. Towards Model Transformation
3. Translation Using Acceleo
4. Conclusion and Perspectives

Model transformation techniques

- Easy transformation : model-to-model techniques
  - Requires metamodels
  - Metamodel for SMDs: OK [OMG, 2011]
  - But absence of coloured Petri nets metamodel

[1] http://www.eclipse.org/acceleo/

Model transformation techniques

- Easy transformation : model-to-model techniques
  - Requires metamodels
  - Metamodel for SMDs: OK [OMG, 2011]
  - But absence of coloured Petri nets metamodel
- Use of model-to-text techniques
  - Requires only the source metamodel (UML)
  - Implementation with Acceleo[1]

[1] http://www.eclipse.org/acceleo/

Acceleo

- Tool based on model-to-text techniques
- Takes as input the source metamodel, and a model compliant with that metamodel
    - Defined using EMF
- User-friendly: Eclipse plugin
- Generated text mixed with Acceleo syntax

Outline

1. Concepts
2. Towards Model Transformation
3. Translation Using Acceleo
4. Conclusion and Perspectives

Structure of the translation

Translation based on [A., Choppy, Klai, 2012]

Based on three algorithms:

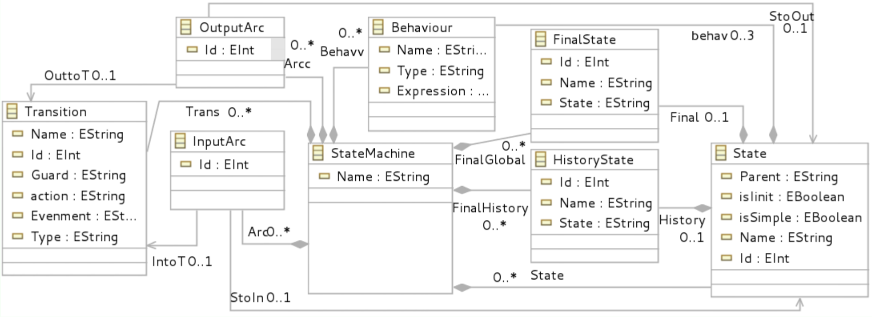
- Translation of states
- Translation of transitions
- Translation of history pseudostates

Generation of an input model for CPN Tools [Westergaard, 2013]

Principle of the implementation: Metamodel

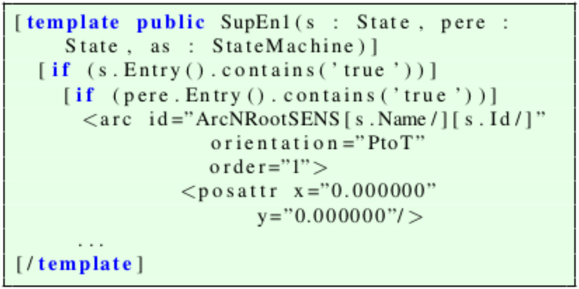
Acceleo needs a metamodel of the source formalism

Simplification (and small adaptation) of the OMG model



Advantages of Acceleo

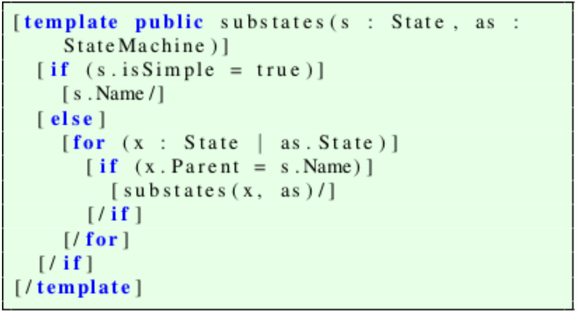
- User-friendly: Eclipse plugin
- No need for a coloured Petri nets metamodel
- The mixed text/code allows us to directly generate a CPN tools input model



```
[template public SupEn1(s : State, pere :
    State, as : StateMachine)]
  [if (s.Entry().contains(’true’))]
    [if (pere.Entry().contains(’true’))]
      <arc id=”ArcNRootSENS[s.Name/][s.Id/]”
                orientation=”PtoT”
                order=”1”>
             <posattr x=”0.000000”
                  y=”0.000000”/>
      ...
[/template]
```

Limitations of Acceleo

- Absence of variable declarations and data structures
- Absence of functions: problem for defining recursivity



```
[template public substates(s : State, as :
    StateMachine)]
  [if (s.isSimple = true)]
    [s.Name/]
  [else]
    [for (x : State | as.State)]
      [if (x.Parent = s.Name)]
        [substates(x, as)/]
      [/if]
    [/for]
  [/if]
[/template]
```

Outline

1. Concepts
2. Towards Model Transformation
3. Translation Using Acceleo
4. Conclusion and Perspectives

Conclusion

- Elaboration of tool UML2CPN for an automatic translation
  - Current state: functional but third algorithm missing
  - Relatively efficient. . . surprisingly!
- Resolution of Acceleo limitations using tips (or “hacks”)
- Acceleo not perfect for this kind of translation
  - Problem of maturity?

Perspectives

- Create a new home-made tool adapted to the translation
- Simplification of the resulting coloured Petri net (including the functions)
- Comparison of our translation with existing semantics for CPNs and SMDs [Liu et al., 2013]
- Integration of timed events

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.

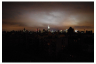
References II

Westergaard, M. (2013).
CPN Tools 4: Multi-formalism and extensibility.
In *Petri Nets*, volume 7927 of *Lecture Notes in Computer Science*, pages 400–409. Springer.

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)

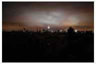
Allusion to any plane crash
(Picture actually from the happy-ending US Airways Flight 1549, 2009)

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

Authors: Étienne André and Mohamed Mahdi Benmoussa