A Modular Approach for Reusing Formalisms in Verification Tools of Concurrent Systems

Étienne André, Benoît Barbot, Clément Démoulins, Lom Hillah, Francis Hulin-Hubard, Fabrice Kordon, Alban Linard, Laure Petrucci

ENS Cachan, Université Paris 6, Université Paris 13, Epita (France)

31 October 2013

CosyVerif

Motivation

Dissemination of Verification Tools

- Application of formal methods to dedicated cases studies
- Towards technological transfer to industry
- Tools organised around formalisms

Academics $\neq$ Developers 

- Need to share effort (platform, interfaces, distribution mechanisms)
- Need to share definitions (typically formalisms)
- Coordinated effort to better handle a complex context of interrelated formal notations
  - *Variants of Petri nets*
  - *Variants of automata*
  - *etc.*

Outline

Outline

A Unified Representation

CosyVerif, a new integration platform [AHHKLLP13]

- Based on Web services
- Easy integration of tools (into services)

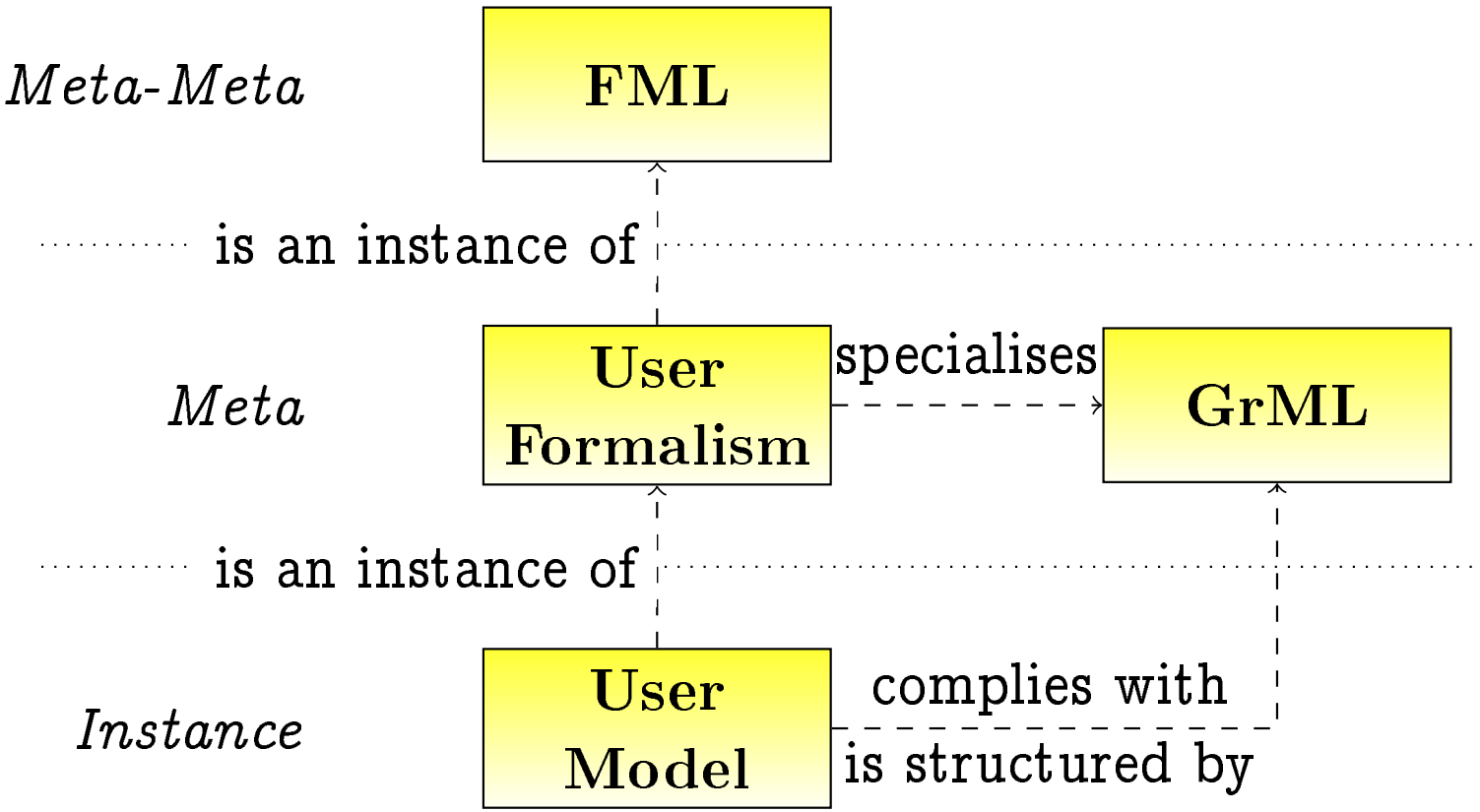
Needs to define formalisms

- Easy definition and share of formalisms

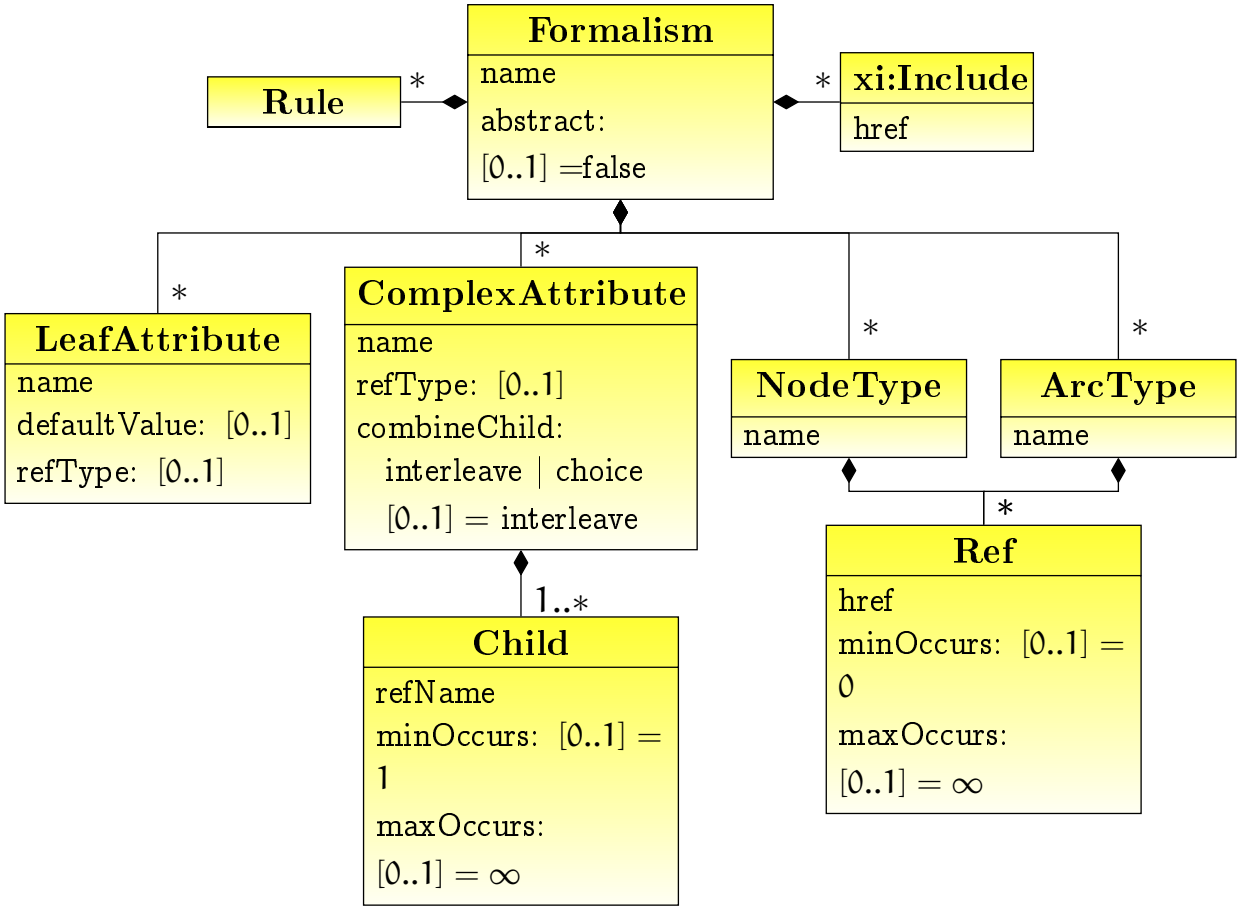
- It is crucial to define formalisms in an easy and sustainable way
- It is crucial to share portions of formalisms (e.g. inheritance)
- It is crucial to combine portions of formalisms (e.g. modularity and hierarchy)

A Unified Representation

A two-layered XML-based modelling language



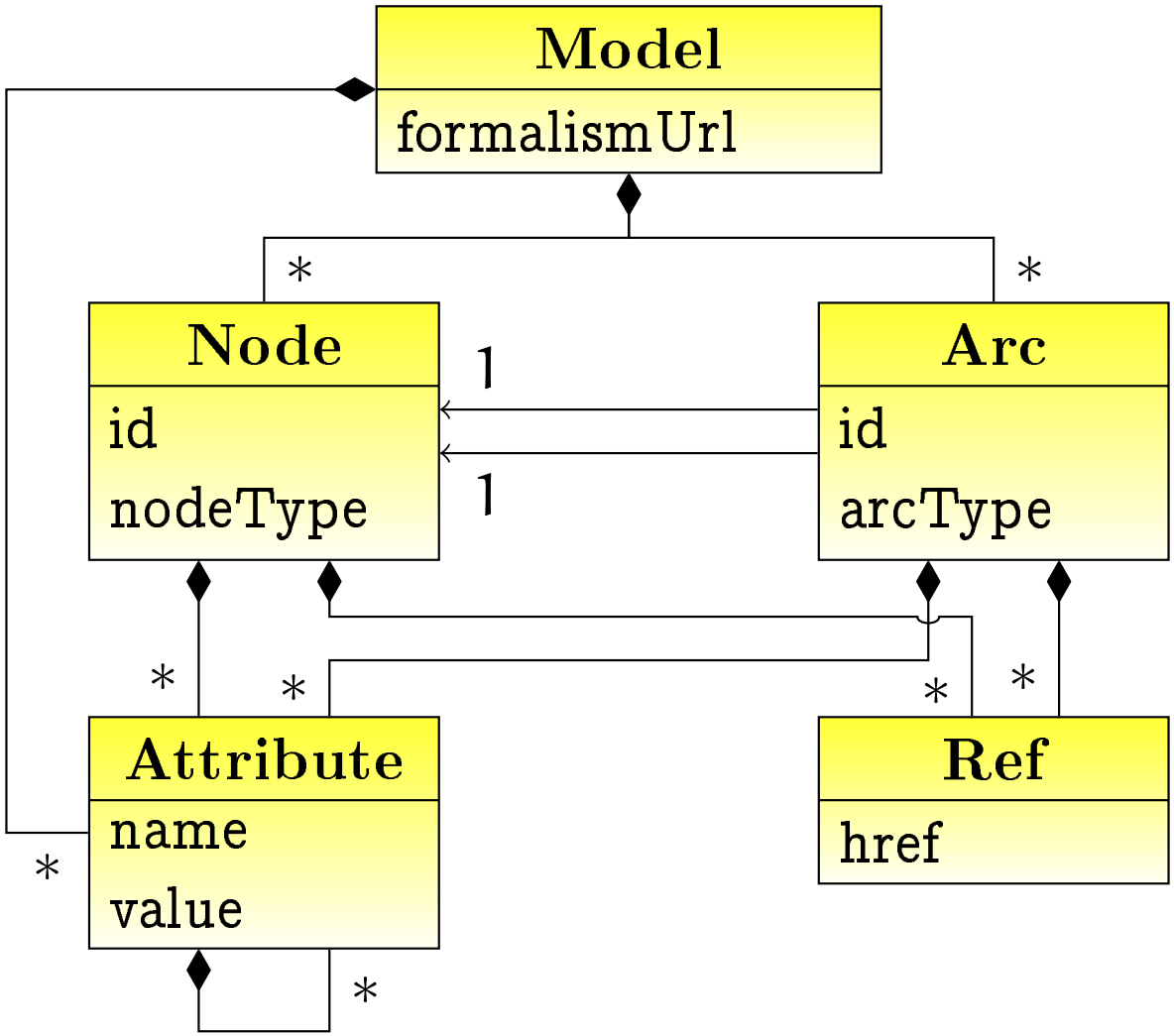
FML: Formalism Markup Language



Example: The Automata Formalism

```
<formalism name="Automaton" xmlns="http://cosyverif.org/ns/
    formalism">
    <leafAttribute name="initialState" />
    <leafAttribute name="finalState" />
    <complexAttribute name="type" refType="state">
        <child refName="initialState" minOccurs="0" maxOccurs="1"/>
        <child refName="finalState" minOccurs="0" maxOccurs="1"/>
    </complexAttribute>
    <leafAttribute name="name" refType="state"/>
    <leafAttribute name="label" refType="transition"/>
    <nodeType name="state"/>
    <arcType name="transition"/>
</formalism>
```

GrML: Graph Markup Language

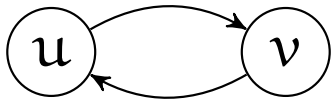


Example: An Automaton

Example of a GrML code:

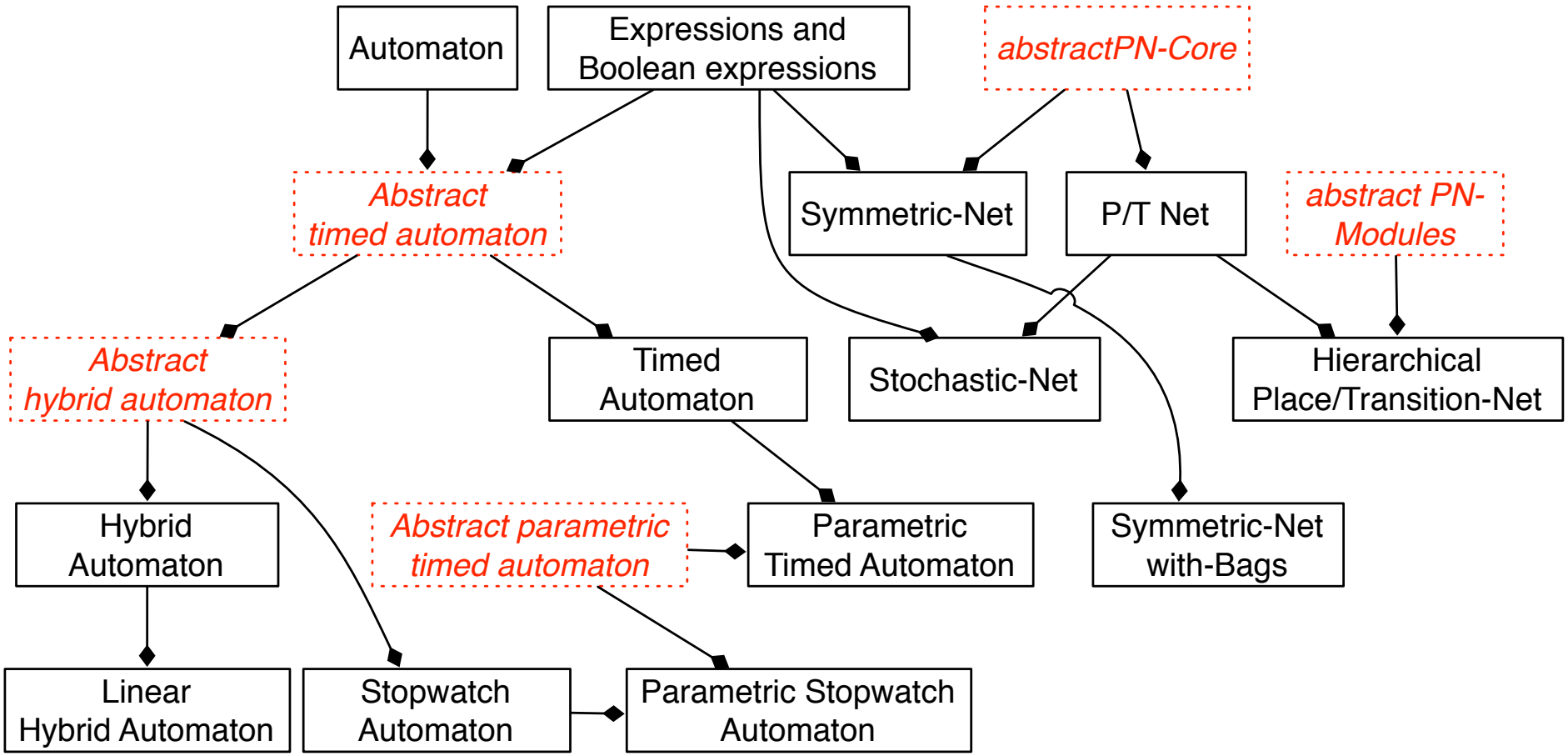
```
<?xml version="1.0" encoding="UTF-8"?>
<model formalismUrl="http://formalisms.cosyverif.org/graph.fml"
xmlns="http://cosyverif.org/ns/model">
<node id="1" nodeType="vertex">
<attribute name="name">u</attribute>
</node>
<node id="2" nodeType="vertex">
<attribute name="name">v</attribute>
</node>
<arc id="101" arcType="transition" source="1" target="2"/>
<arc id="102" arcType="transition" source="2" target="1"/>
</model>
```

Corresponding automaton:



Architecture of Formalisms

Application to Automata and Petri Nets



Automated Compliance Checking

- GrML and FML syntaxes validated using RELAX-NG
- Compliance of a GrML model against its FML formalism
  - *Schematron rules derived from the XML description of the formalism*
  - *Particular constraints such as "in a Petri net formalism, no arc should connect two nodes of the same type"*
  - *Implemented in GrML-Check*

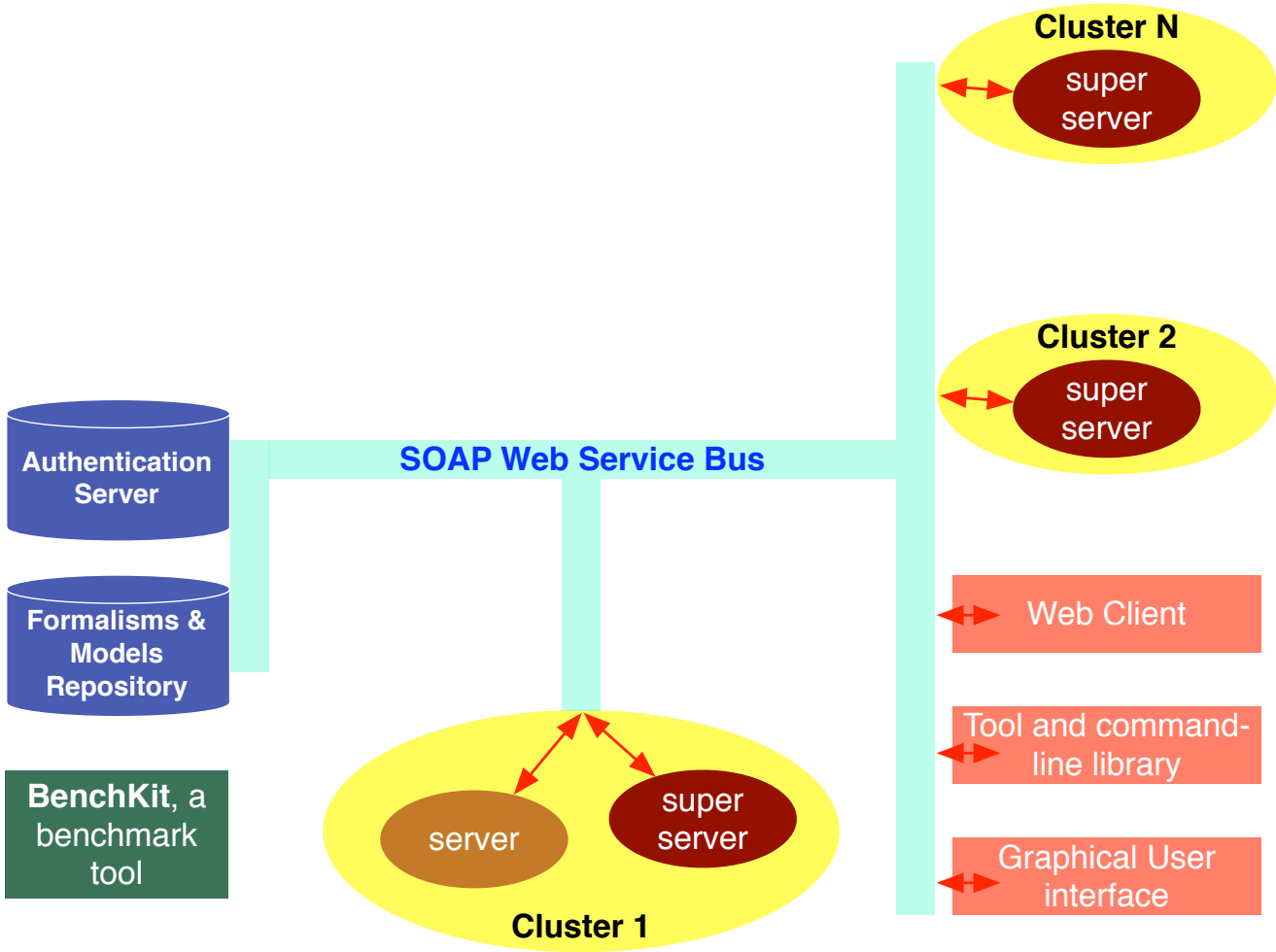
Good Practices for Defining Formalisms

- Abstract vs. concrete formalisms
  - *Similarity with object-oriented paradigm*
  - *Definite concrete formalisms as late as possible in the hierarchy*
  - *Only concrete formalisms can be instantiated*
- Separate and reuse
  - *Separate data types (integers, colors, etc.)*
  - *Example in our hierarchy: Expressions and Boolean expression*
- Progress by small increments
  - *P/T nets extended to Stochastic nets*

Outline

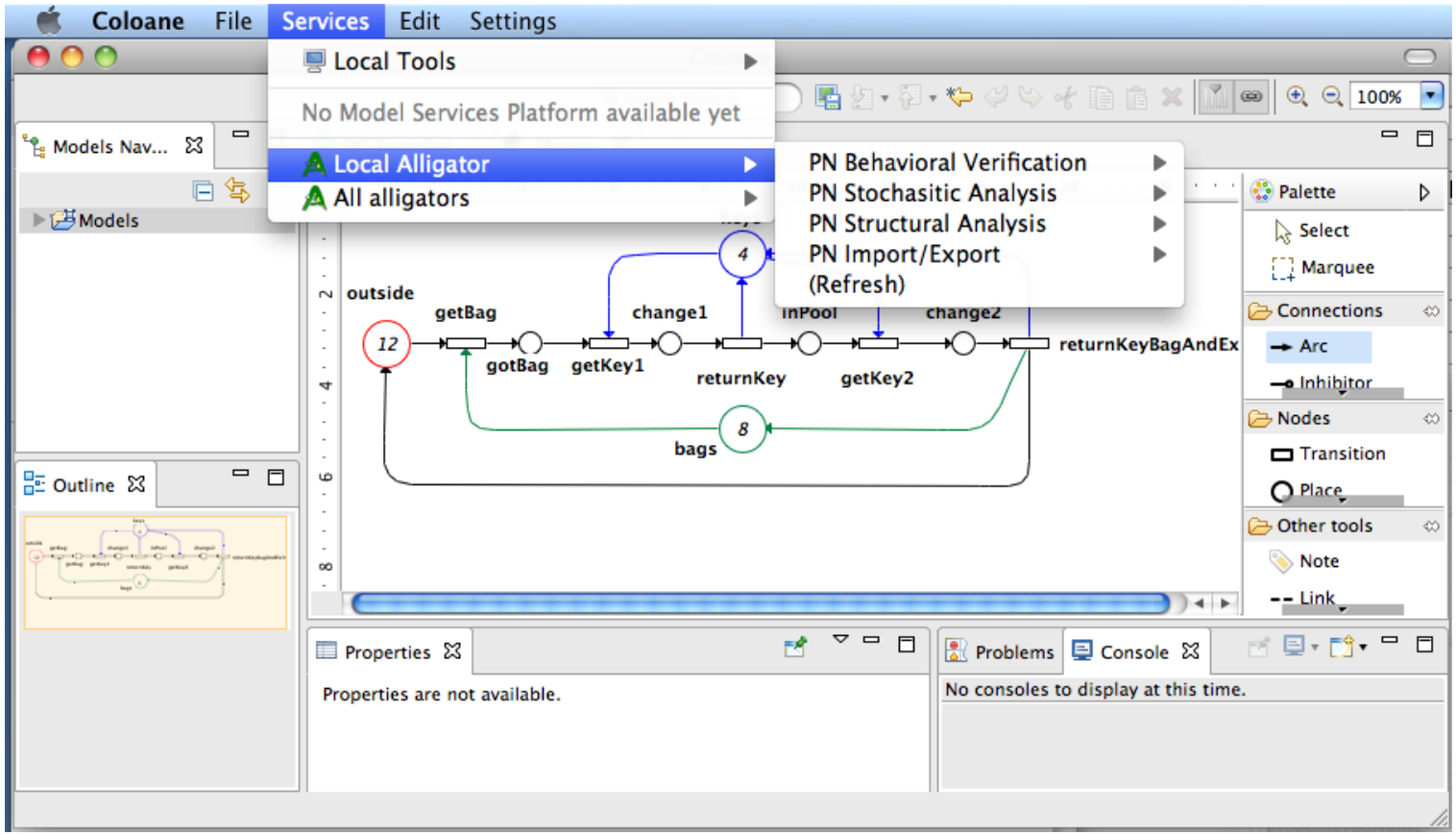
The *CosyVerif* Verification Platform

The *CosyVerif* Architecture



The *Coloane* User Interface

The *Coloane* User interface



Command-line client is also available for script-based access to tools

Formalisms and Tools

Formalisms	Tools	
Petri Nets	PROD (Univ. Helsinki, Symmetric nets)	
	PNXDD (LIP6, Symmetric nets)	[Hong et al., 2012]
	Crocodile (LIP6, Symmetric nets w. bags)	[Colange et al., 2011]
	Cunf (LSV, P/T nets)	[Baldan et al., 2012]
	Cosmos (LSV, Stochastic Petri nets)	[Ballarini et al., 2011]
	GreatSPN invariants (Univ. Torino, P/T nets)	
	Structural bounds (LIP6, P/T nets)	
	Unfold into P/T nets (LIP6, Symmetric nets)	
	Various exports (LIP6, P/T nets)	
Automata	IMITATOR (LIPN, Timed automata)	[André et al., 2012]
	Modgraph (LIPN, Synchronised automata)	[Lakos and Petrucci, 2004]

Download and Try!

Two bundles available (`http://download.cosyverif.org`)

All services   Petri Nets

Entirely open source (GPL/AGPL)

- Server
- Client(s)
- Bundles
- Tools

Download and Try!

Two bundles available (http://download.cosyverif.org)

All services   Petri Nets

Entirely open source (GPL/AGPL)

- Server
- Client(s)
- Bundles
- Tools

Try it!

http://www.cosyverif.org/

Outline

Conclusion & Perspectives

Summary

- A unified representation of formalisms (FML)
- A description of models (GrML)
- Automated conformance checks
- Implementation in *CosyVerif*

Perspectives

- Formalism for properties
- Extension of FML to define composition
- Semantics
  - *Heterogeneous model composition*
  - *Automated translation from formalisms to other formalisms*

Bibliography

References I

André, É., Fribourg, L., Kühne, U., and Soulat, R. (2012).
IMITATOR 2.5: A tool for analyzing robustness in scheduling problems.
In *FM*, volume 7436 of *Lecture Notes in Computer Science*, pages 33–36. Springer.

André, É., Hillah, L.-M., Hulin-Hubard, F., Kordon, F., Lembachar, Y., Linard, A., and Petrucci, L. (2013).
CosyVerif: An open source extensible verification environment.
In Liu, Y. and Martin, A., editors, *18th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS'13)*, pages 33–36. IEEE Computer Society.

Baldan, P., Bruni, A., Corradini, A., König, B., Rodríguez, C., and Schwoon, S. (2012).
Efficient unfolding of contextual Petri nets.
Theoretical Computer Science, 449:2–22.

Ballarini, P., Djafri, H., Duflot, M., Haddad, S., and Pekergin, N. (2011).
HASL: An expressive language for statistical verification of stochastic models.
In *VALUETOOLS*, pages 306–315.

References II

Colange, M., Baarir, S., Kordon, F., and Thierry-Mieg, Y. (2011).
Crocodile: A symbolic/symbolic tool for the analysis of symmetric nets with bags.
In *ICATPN*, volume 6709 of *Lecture Notes in Computer Science*, pages 338–347. Springer.

Hong, S., Kordon, F., Paviot-Adet, E., and Evangelista, S. (2012).
Computing a Hierarchical Static order for Decision Diagram-Based Representation from P/T Nets.
Transactions on Petri Nets and Other Models of Concurrency (ToPNoC), V:121–140.

Lakos, C. and Petrucci, L. (2004).
Modular analysis of systems composed of semiautonomous subsystems.
In *ACSD*, pages 185–196. IEEE Computer Society.

Licensing

License of this document

Author: The *CosyVerif* team