A Survey on Multi-Formalism Performance Evaluation Tools

Simonetta Balsamo Gian-Luca Dei Rossi Andrea Marin

Dipartimento di Scienze Ambientali, Informatica e Statistica
Università Ca' Foscari, Venezia

ESM '12, Essen, 22-24 October 2012

Motivation

- Performance and reliability evaluation is useful
  - widely studied topic
  - powerful tool for system designers and maintainers
  - different formalisms and different solution techniques
- Modelling phase is difficult
  - Formalisms require a specific knowledge
  - Systems should be modelled component-wise
  - Different formalisms for different problems and kind of components
- Use of tools that support multiple formalisms and their composition.

Toy Example

- System where N_u identical users follow a strict corporate workflow
- in some of the workflow phases they request several computations to a set on N_s servers
  - parallel processing of the jobs
- The server are made of many software components which in turn use various (even shared) hardware resources
- After all the computations are completed, the users spend time to merge them, and produces an output.

What if we want to compute some *performance indices* on this system?

- Which formalism is better suited for this task?

What we consider and what we don't

- A survey on tools that support multiple formalisms.
- Classification criteria
  - Formalisms
  - Solutions
  - User interface
  - Documentation
  - License
  - Maintenance status
  - Supported platforms
- What we didn't analyse
  - Interoperability
  - Ease of use
  - Pricing scheme
  - ...

What we consider and what we don't

- Wide plethora of tools for different formalism families
  - Markov Processes
  - Stochastic or Probabilistic Process Algebras
  - Queueing Networks
  - Stochastic Petri Nets
  - ...
- We considered only tools for which we were able to verify the existence of an actual implementation
  - Active development after the year 2000
  - Multiple references in published papers
  - Website
- We identified 4 tools which satisfy the conditions: **SHARPE**, **Möbius**, **SmArT**, **SIMTHESysEr**

SHARPE

- **S**ymbolic **H**ierarchical **A**utomated **R**eliability and **P**erformance **E**valuator
- Developed at Duke University
- Interactive or unattended (scripted) use
- Graphical User Interface
- Formalisms: Markov and semi-Markov chains, Markov regenerative processes, Multi-Chain Product Form Queueing Network, Generalised Stochastic Petri Nets , Stochastic Reward Nets, Reliability Block Diagrams, Fault Trees, Reliability Graphs, Series-Parallel Graphs
- Numerical solutions for steady-state and transient analysis.
- *Hierarchical* multiformalism composition
- Books

Möbius

- **S**ymbolic **H**ierarchical **A**utomated **R**eliability and **P**erformance **E**valuator
- Developed at the University of Illinois
- Interactive or unattended (scripted) use
- Graphical User Interface
- Formalisms: Stochastic Activity Networks, Buckets and Balls, PEPA$_k$, Fault Trees
- Exact Numerical solutions (when available) and simulation.
- Distributed computation
- *Flat* multiformalism composition

SmArT

- **S**tochastic **M**odel checking **A**nalyzer for **R**eliability and **T**iming
- Developed at the University of California, Riverside
- Unattended (scripted) use through an ad hoc programming language
- Formalisms: Markov Chains, Stochastic Petri Nets
- Numerical solutions for steady-state and transient analysis.
- Model checking

SIMTHESysEr

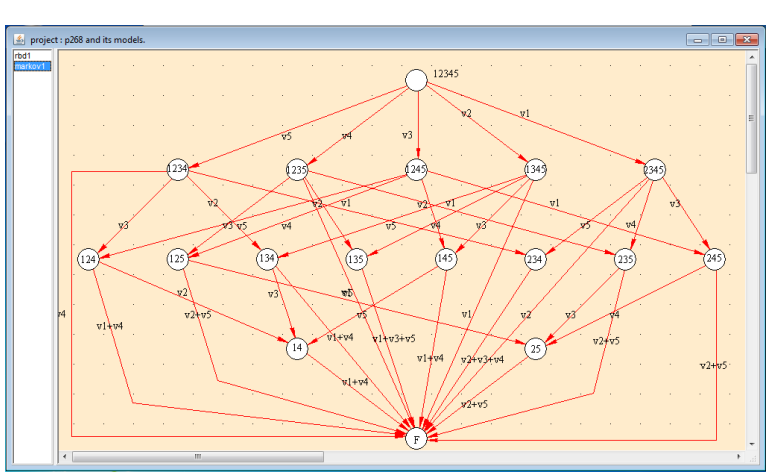
- Based on the SIMTHESys approach to modelling
  - **S**tructured **I**nfrastructure for **M**ultiformalism modelling and **T**esting of **H**eterogeneous formalisms and **E**xtensions for **SYS**tems
- Developed at 3 Italian universities
- General framework to develop new tools
- User defined classes of models and solvers
  - Pluggable interaction among modules
- Some already implemented formalisms
- Some already implemented solvers

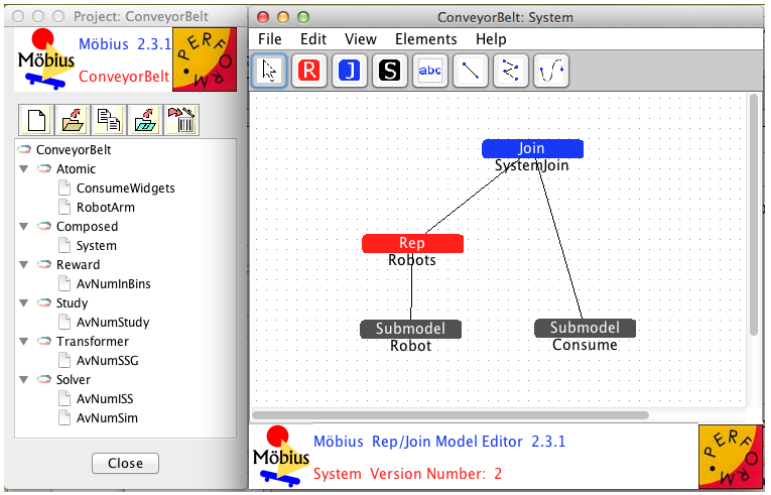
Comparison: formalisms and solutions

- SHARPE offers the widest choice of formalisms.
  - not all formalisms can be used in a hierarchy of submodels
  - Numerical solutions
- Möbius is the only one to support PEPA.
  - All formalisms can be used in composed models
  - Numerical solutions or (distributed) simulations
- SIMTHESysEr could be extended to support arbitrary formalisms and solution methods
- SmArT offers two families of formalisms
  - numerical solutions
  - we can do *model checking* on them!

Comparison: user interfaces

- SHARPE and Möbius have graphical user interfaces





- SIMTHESysEr can parse models designed using Draw-Net
- SmArT can be used non-interactively through a programming language

Comparison: documentation/maintenance/license

- SHARPE
  - proprietary, free for academic users
  - actively maintained
  - well documented
- Möbius
  - proprietary, free for academic users
  - actively maintained
  - very well documented
- SmArT
  - proprietary, licensing not disclosed
  - no recent updates
  - well documented
- SIMTHESysEr
  - freely available source code
  - in ongoing development
  - less documented

Comparison

	SHARPE	**Möbius**	**SmArT**	**SIMTHESysEr**
Supported Formalisms	Markov and Semi-Markov, RBDs, FTs, RGs, MCPFQN, GSPNs, SPGs	AN, Buckets and Balls, PEPA$_k$, Fault Trees	Markov Chains, SPNs	pluggable, ATM Markov Chains, QNs (limited), SPNs
Solution Methods	Numerical (Approximate and Exact)	Exact Numerical, Simulation	Numerical	pluggable, ATM CTMC solution and simulation
User Interface	Textual, GUI	Textual, GUI	Textual	Textual
Platforms	Windows (Linux and Solaris in older versions)	Linux, Mac-OSX, Windows	Linux, Mac-OSX, Windows	Windows (other platforms may work)
License	Proprietary, no cost for academic users	Proprietary, no cost for academic users	Not Specified (on demand)	Source code freely available
Maintenance	Supported and updated	Supported and updated	Not recently updated	In active development

Conclusion

- All the analysed softwares are powerful tools for modelling complex systems
- The choice depends on the users' requirements
  - Professionals: implementation polishing, commercial support, GUIs.
  - Academics: source code availability, extendability, books
  - All users: documentation
- Single-formalism tools could be better suited for specific tasks
- The intrinsic difficulties in building multi-formalism tools limit the number of available software packages
  - Academic institutions could not have resources to maintain the code.
- Possible future works: quantitative analysis on the performances of the tools themselves.

Thanks!

Thanks for the attention

any question?