

Multiformalism modeling

An introduction to multiformalism modeling for performance evaluation

Prof. Ing. Mauro Iacono, PhD.

President of the European Council for Modelling and Simulation

Director of the ECMS International School for Young Researchers

1st ECMS International School
for Young Researchers
Politechnika Krakowska, 3/6/2024

 Università
degli Studi
della Campania
Luigi Vanvitelli



What will we do today?

- What is performance modeling? Why do we use it?
- Examples: Queuing Networks, Stochastic Petri Nets
- Multiformalism modeling

Performance modeling

- *Performance Evaluation* is the quantitative and qualitative study of systems, to evaluate, measure, predict and ensure target behaviors and performances
- It is usually carried on using *models of a system*
- A model is an abstraction of a system:

"an attempt to distill, from the details of the system, exactly those aspects that are essentials to the system behavior"....

(E. Lazoswka)

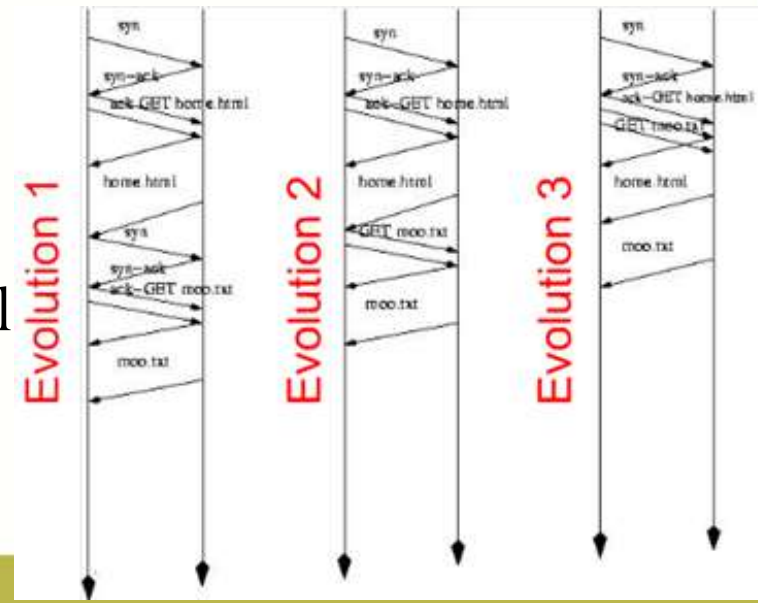
- After a model has been defined, it is usually exploited in three steps:
 - Validation
 - Projection
 - Verification

Performance modeling

- During the *validation* phase, results predicted by the model are compared against measurement of the real system to check if they match
- The *projection* changes some of the parameters of the model (i.e. the speed or the quantity of a component) and computes the corresponding indices to see if a goal is met in the new configuration
- *Verification* actually modifies the real system according to the new parameters tested in the model, to check whether the objective has been really achieved

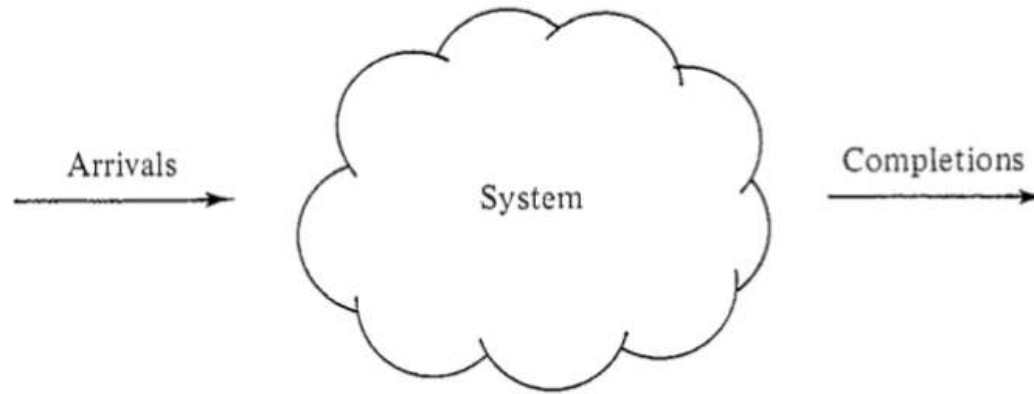
Solution techniques

- Once a model has been defined, its performance indices are computed using suitable solution techniques. Different procedures exist, and the most common are:
 - *Analytical* and *numerical* techniques are based on the application of mathematical techniques, which usually exploit results coming from the theory of probability and stochastic process
 - If the technique uses expressions in closed form (i.e. a formula or an exact algorithm to compute them can be derived), it is said to be *analytical*
 - If the solutions can only be obtained using numerical procedures, the techniques are said to be *numerical*
 - Simulation is based instead on the reproduction of *traces* of the model
 - A trace is a possible sequence of events that can characterize one possible evolution of the model



A problem with arrivals and departures

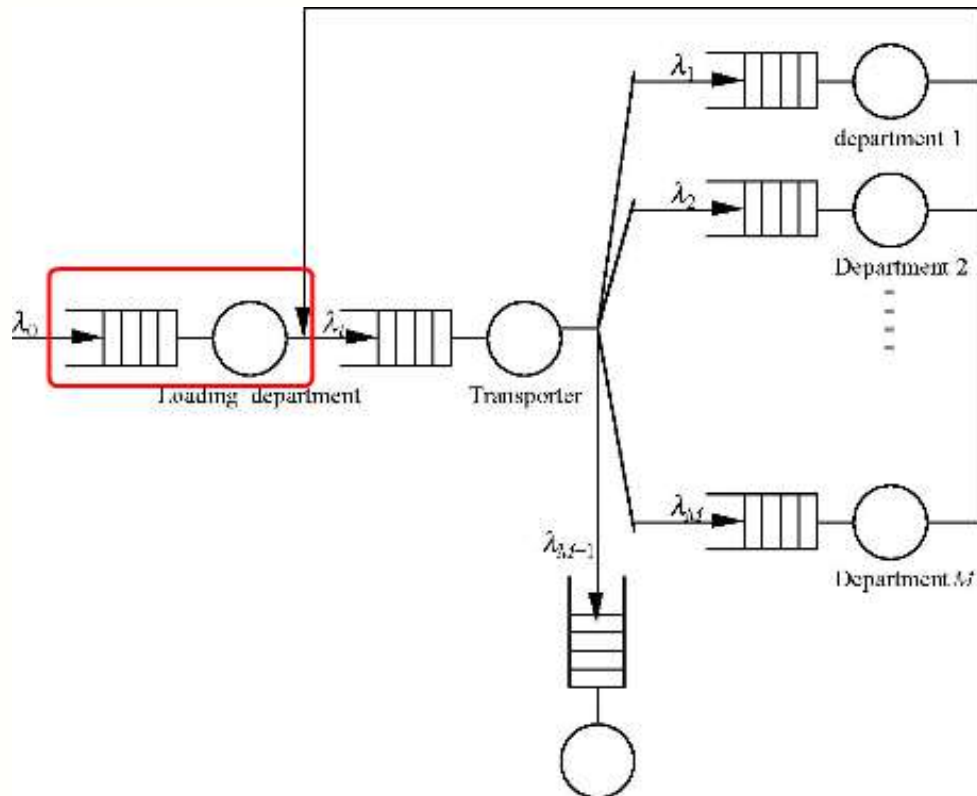
- Let us observe an abstract system, where we can only focus on its entry and exit point, for a given amount of time T



- By simply counting the jobs that enters and leaves the system in the considered time frame, we can determine its main parameters (*arrival rate* and *average service time*), and performance indices (*utilization*, *throughput*, *average service time*, *average number of jobs in the system*)

Example: Queuing Networks

- *Queuing Networks* represent systems using a set of interconnected entities called queues
- Born in the TLC domain, they are widely adopted whenever queuing effects impact on system performances

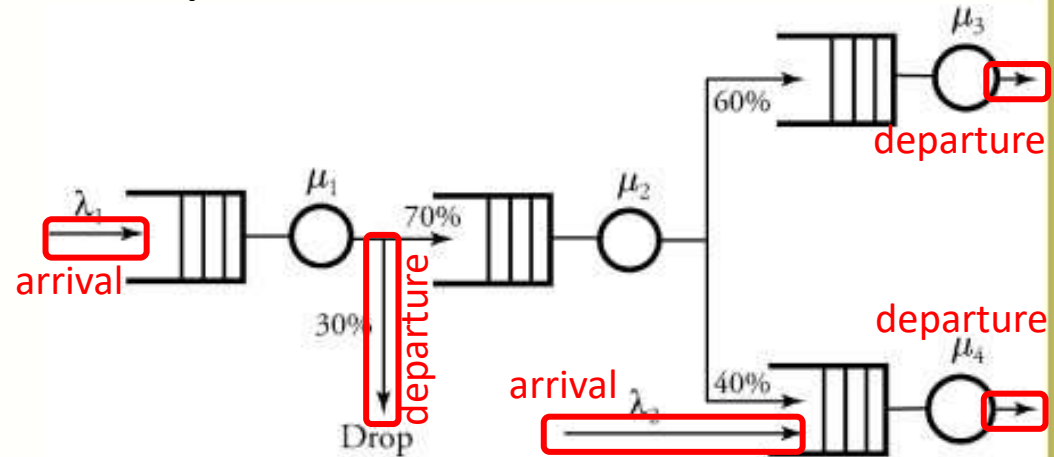


Queuing stations

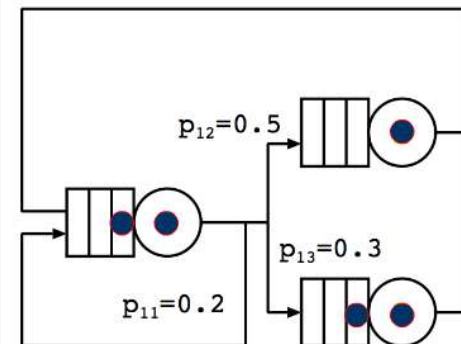
- *Queuing stations* can be used to model several elements of a system:
 - CPUs in multi-tasking S.O., disks, web services, communication channels, routers with buffers
 - Manufacturing machines
 - ...
- Queues are populated by entities that require services
- Depending on the context, such entities are called customers, clients, jobs, tasks, packets, tokens, ...
- Stochastic characterization of *service times* and *arrival rates*
- Evaluation of the stability conditions
- Strong mathematical foundations allow analytical evaluation, event-based simulation as alternative

Open and closed networks

- Queuing networks can be either *open* or *closed*
 - In *open models* jobs arrive from outside at a specified arrival rate
 - In *closed models* there is a fixed population of jobs that moves between the queues inside the system
- *Open Models* are characterized by *arrivals* and *departures* from the system
- In closed models we have a fixed population of N jobs that continuously circulate inside the system



$N=5$



Performance indices

- From a service station, several performance indices can be computed, depending on the problem
- The most important are:
 - the *utilization*: is the fraction of time a server is busy (not waiting for new jobs to arrive)
 - the *response time*: is the average time spent by a job at a service center
 - the *average queue length*: accounts for the mean number of jobs in a service station (both the ones being served and the ones in the queue)
 - the *throughput*: describes the rate at which jobs are served and depart from the station

Network performance indices

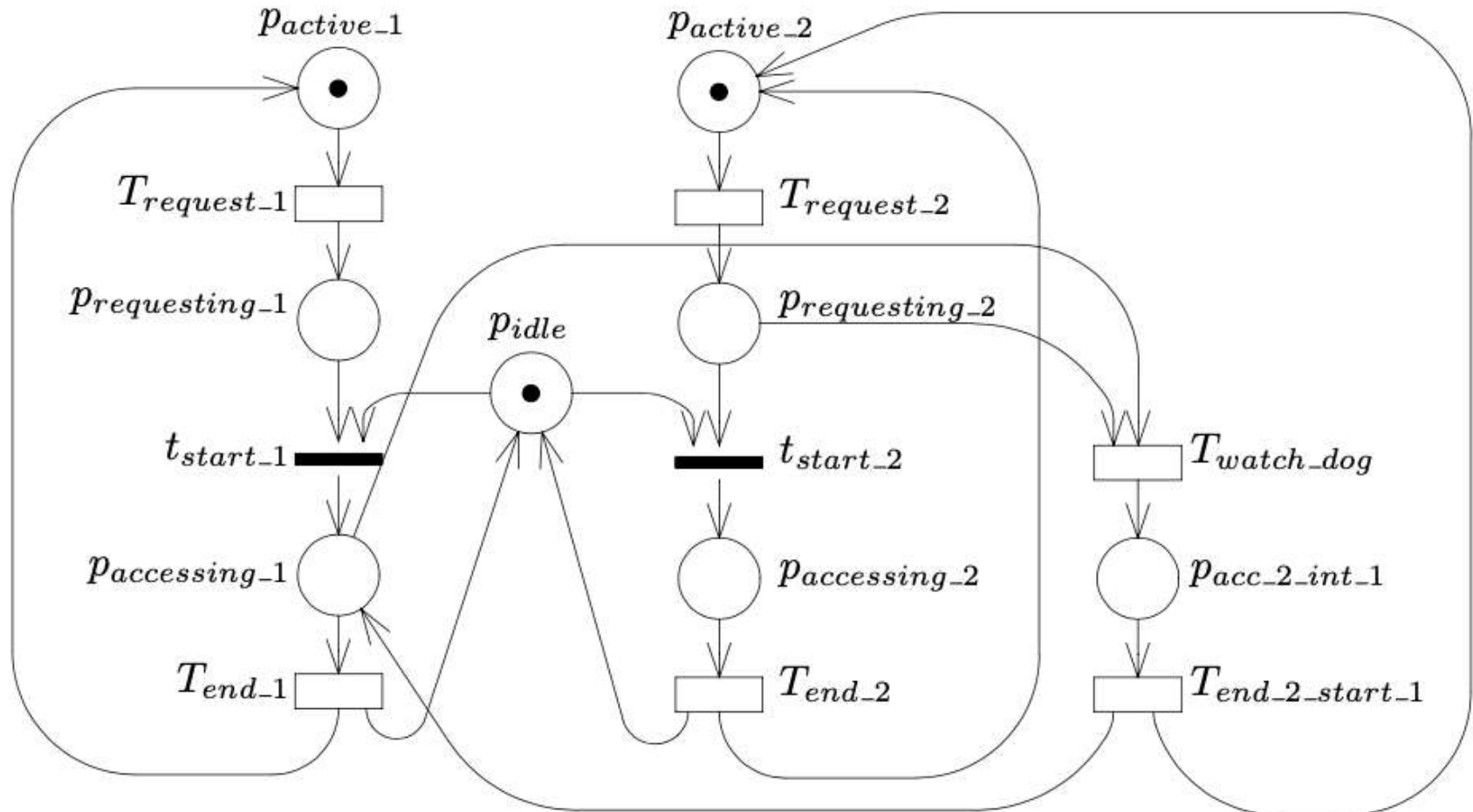
- On network models, extra performance indices can be defined:
 - System throughput
 - Total system population
 - System response time
- For what concerns utilization, there is no unique definition of a system-wise measure:
 - The fraction of time in which there is at least one job in the system
 - The average utilization of all the stations
 - The utilization of the bottleneck station
- All definitions have strength and weaknesses

Example: Stochastic Petri Nets

- Queuing networks are perfect to model systems where jobs are executed through a set of stations
- They are characterized by convenient high-level performance indices such as throughput, response times and utilization
- However, they cannot easily model *resource contentions* and *concurrency*
- Other formalisms, such as *Stochastic Petri Nets*, are used to model systems characterized by such features

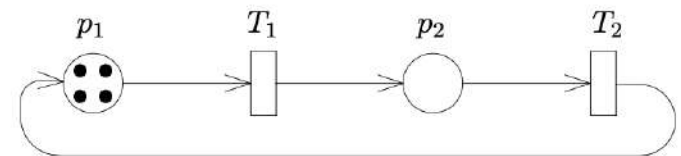
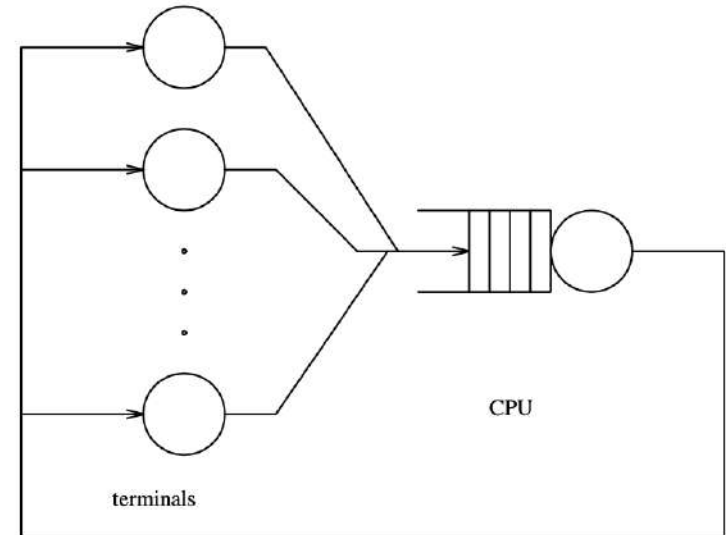
Petri Nets

- Petri Nets are *bi-partite graphs*, characterized by two set of elements: *places* and *transitions*



Petri Nets performance models

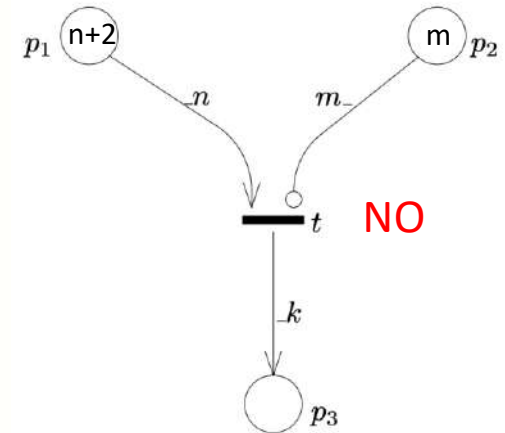
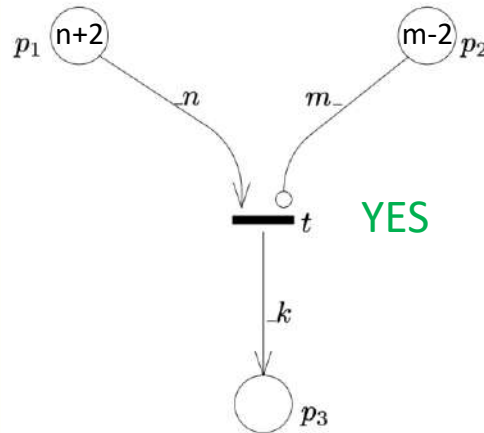
- The meaning of the different type of arcs is the following:
 - *input arcs* model *preconditions*, that must be satisfied for an event to take place
 - *output arcs* are used to specify the *effect* of the actions
 - *inhibitor arcs* prevent an event from taking place
- In performance models created using Petri nets, tokens are used to represent jobs, places to define queues or resource occupancy, and transitions correspond to services



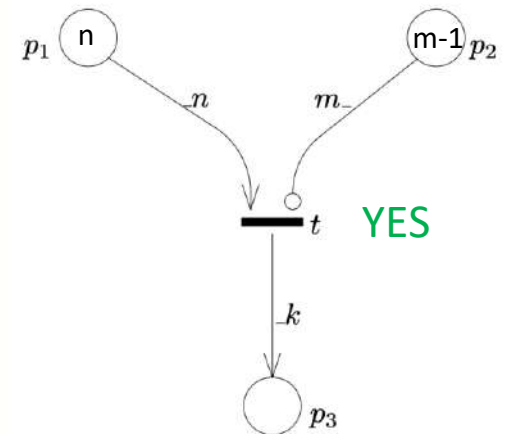
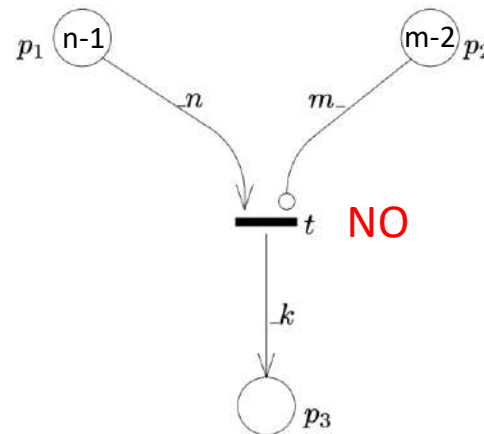
Enabled transitions

- A transition is said to be enabled if:

- Each input place has at least as many tokens as the weight of the corresponding input arc



- Each place connected with an inhibitor arc has less tokens than the weight of the connection



Firing time distribution

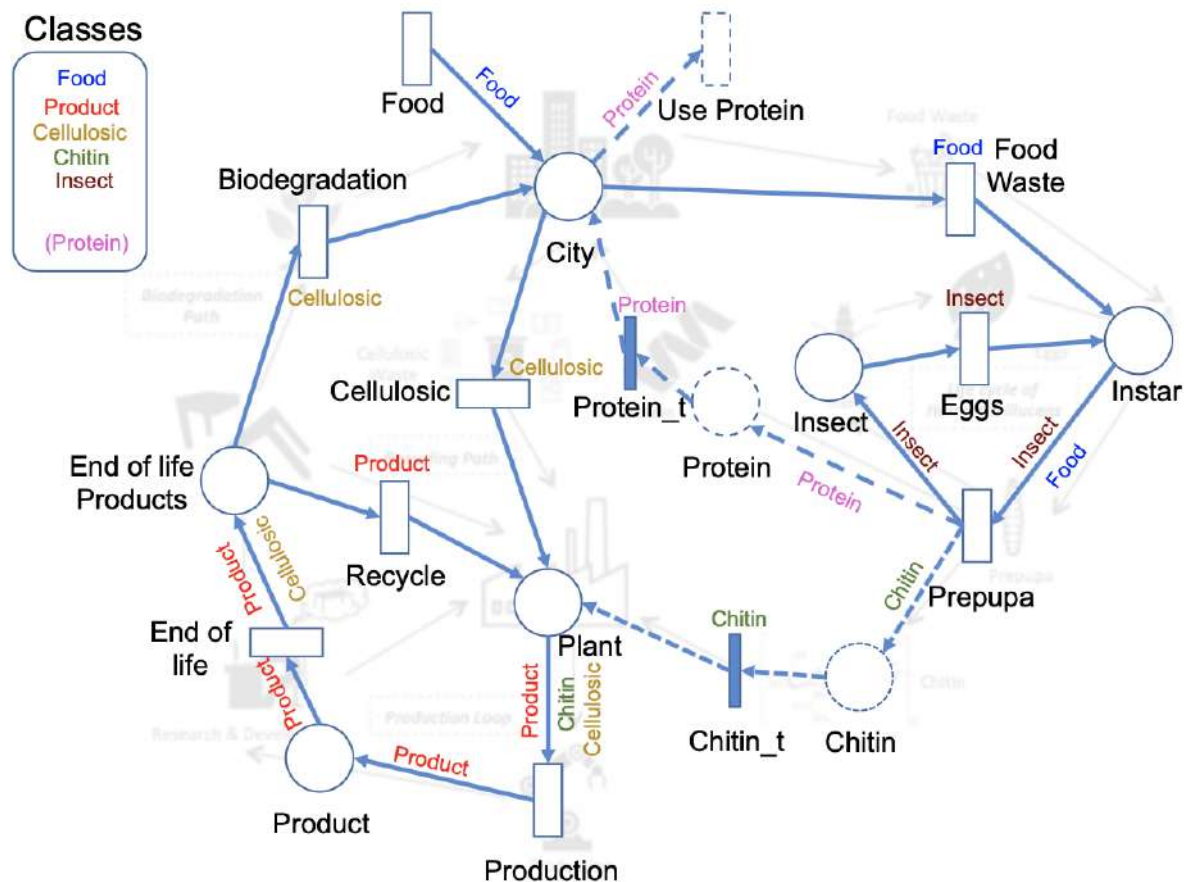
- The original definition of *Stochastic Petri Nets*, associate to each transition an *exponentially distributed random* firing time
- Extended models, called *Non-Markovian Stochastic Petri Nets*, allow to use general firing time distributions
- Conflict are solved with a *race policy*: the system will evolve according to the transition that will fire first

Performance indices: transition throughput

- Only three types of performance indices are defined on PNs
 - the transition throughput counts the average number of firings per time unit done by a transition
 - for places, it is relevant to compute either the probability distribution of having a given number of tokens, or the average number of tokens inside it

Colored Petri Nets

- In *Colored Petri Nets*, tokens are divided into classes called *colors*: each place might contain a different number of tokens for each color



Modeling complex systems

- Problem: modern complex systems have *different aspects* related to *different domains* and *different expertise* but must be represented and evaluated as a whole
- Example: ERTMS/ETCS

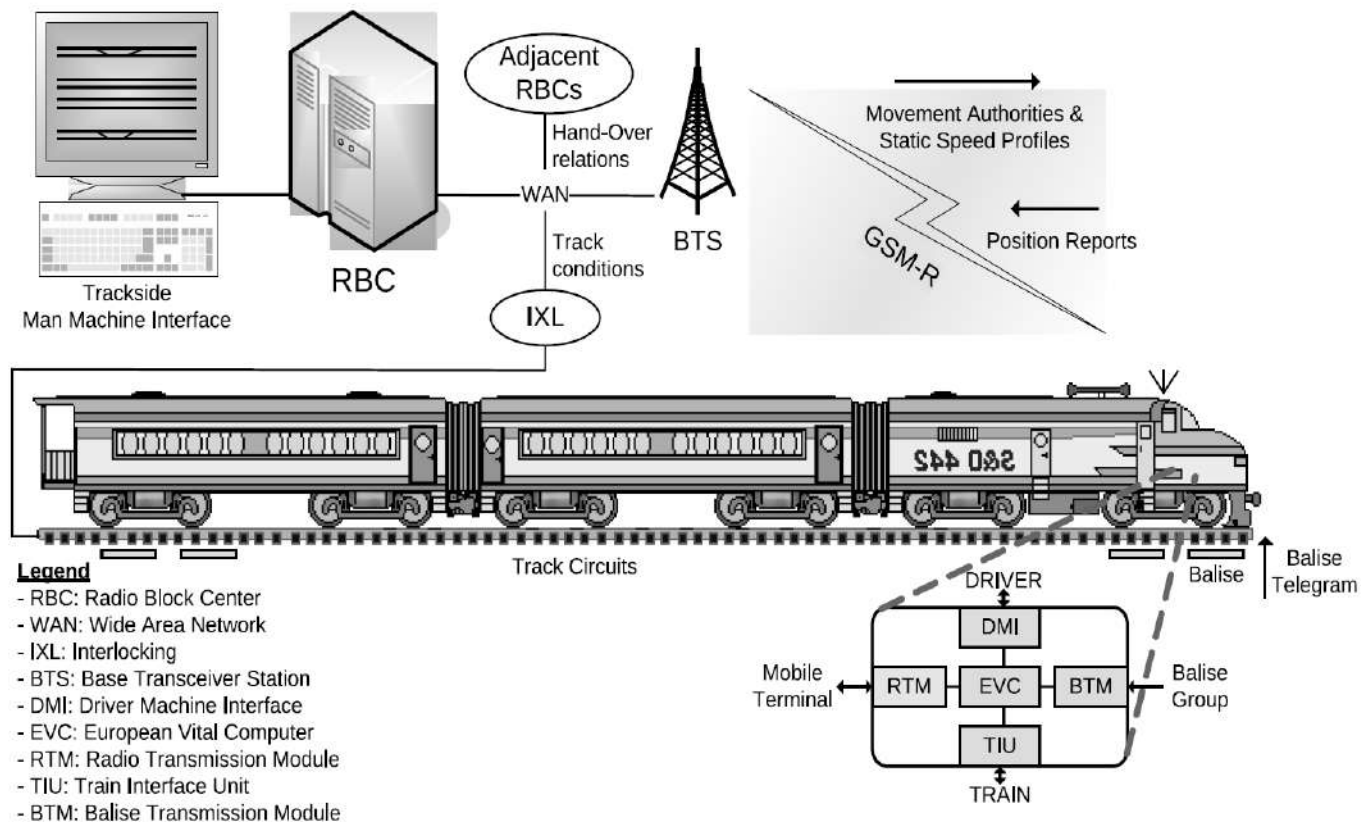


Fig. 4. ERTMS/ETCS system

Multiformalism models

- Multiformalism models allows to exploit different formalisms to describe different components of a system
- They can use the *most appropriate modelling primitive* for each part, and can combine different modeling languages
- One of the main advantages of multi-formalism modelling is its ability to represent the system at *multiple levels of abstraction*
- This allows a *more comprehensive and accurate representation* of the system, which can *facilitate the analysis* and optimization of the behavior of the system

Definition and challenges

- “Multi-formalism modeling is the combination of different formalisms, such as Petri nets, process algebras, and queuing networks, to capture different system behaviours and characteristics”
- Main problems: *semantics, analysis, representation, modularity/compositionality, coherence*

Example of a simple multiformalism model

- One of the most popular example of multiformalism modeling technique is the combination of Queuing Networks and Petri Nets
- If a transition is connected to a queue, whenever the transition fires, it inserts as many jobs as the weight of its arc in the destination queue
- The class of the jobs corresponds to the color of the tokens



Example: architecture

- The lineside subsystem: it is mainly responsible for providing geographical position information to the on-board subsystem;
- The on-board subsystem: it is the core of the control activities located on the train;
- The trackside subsystem: it is in charge of monitoring the movement of the trains.

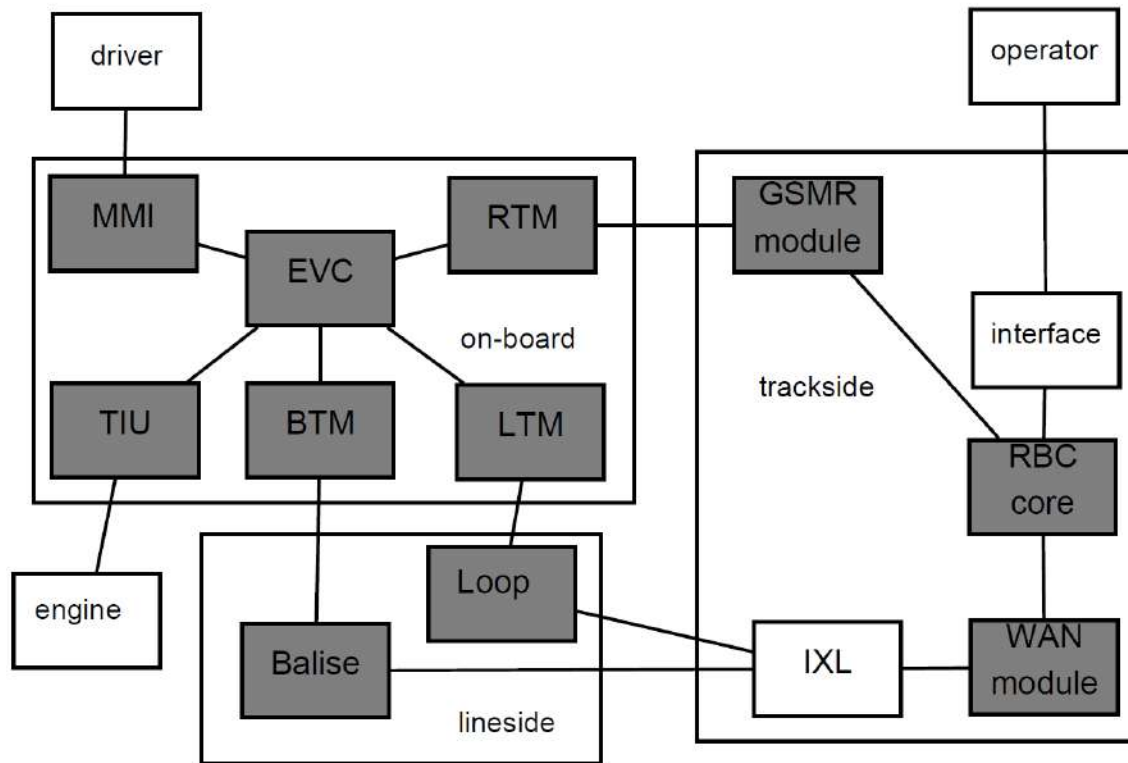


Fig. 1. ERTMS/ETCS architecture

Example: UML component diagrams

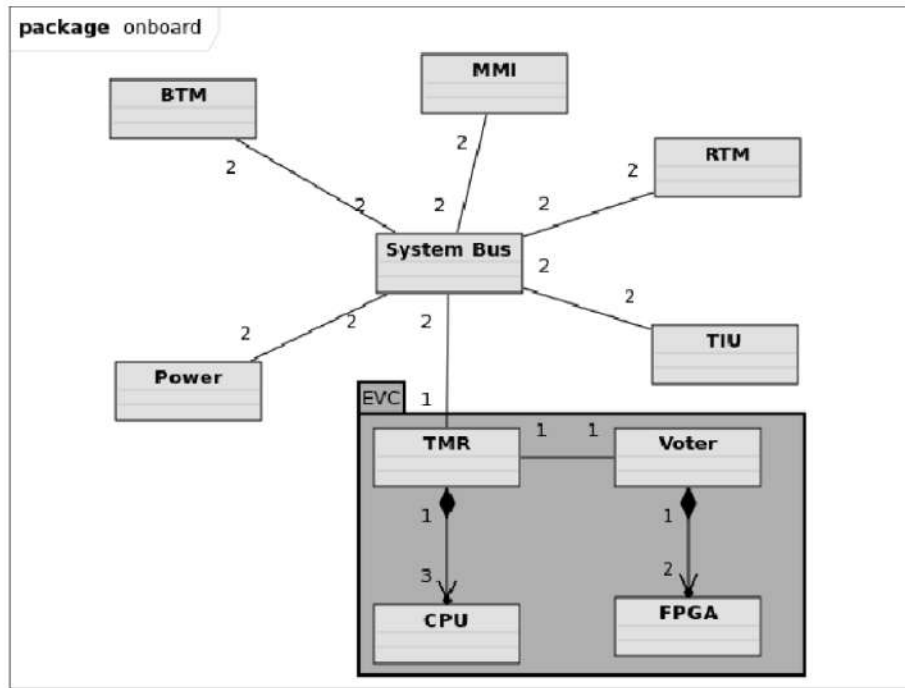


Fig. 2. Class diagram of the ONB subsystem

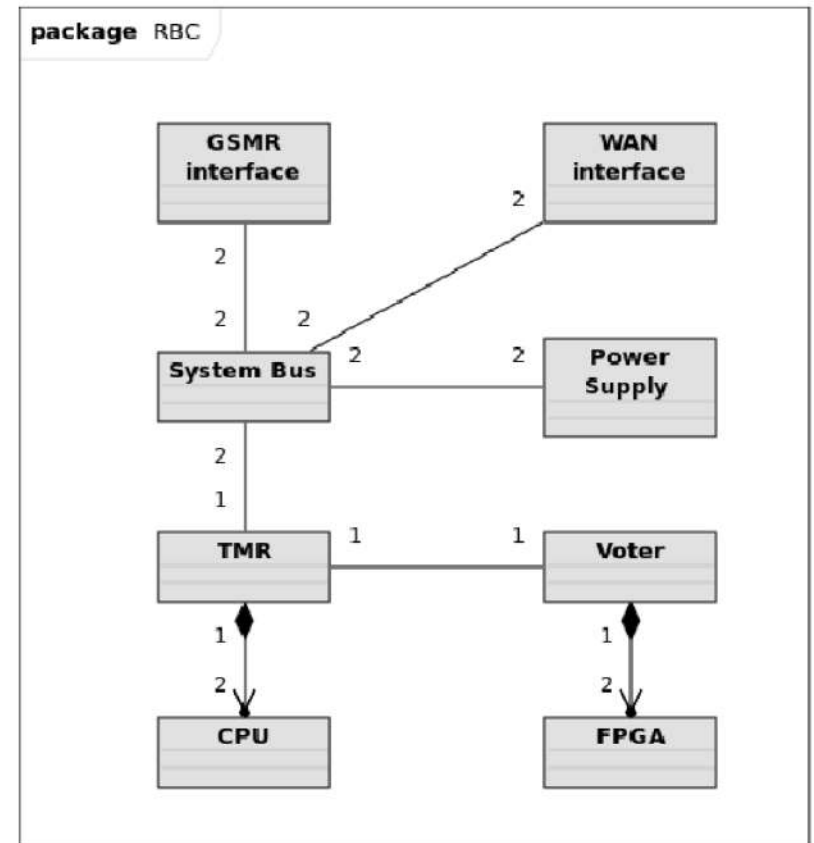


Fig. 3. Class diagram of the RBC component

Example: scenario components

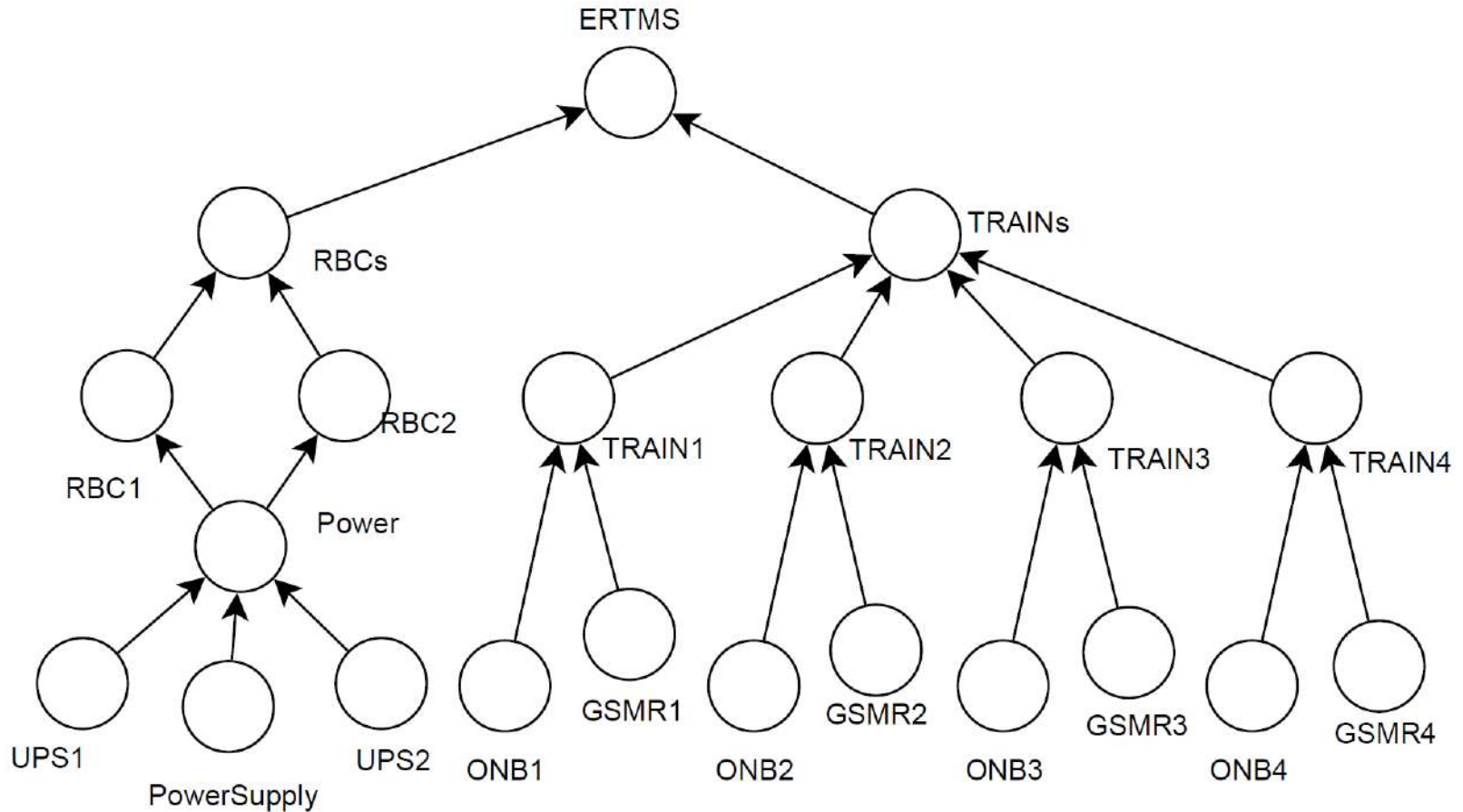


Fig. 5. Contributions of the different sub-models to the overall analysis

Example: reliability

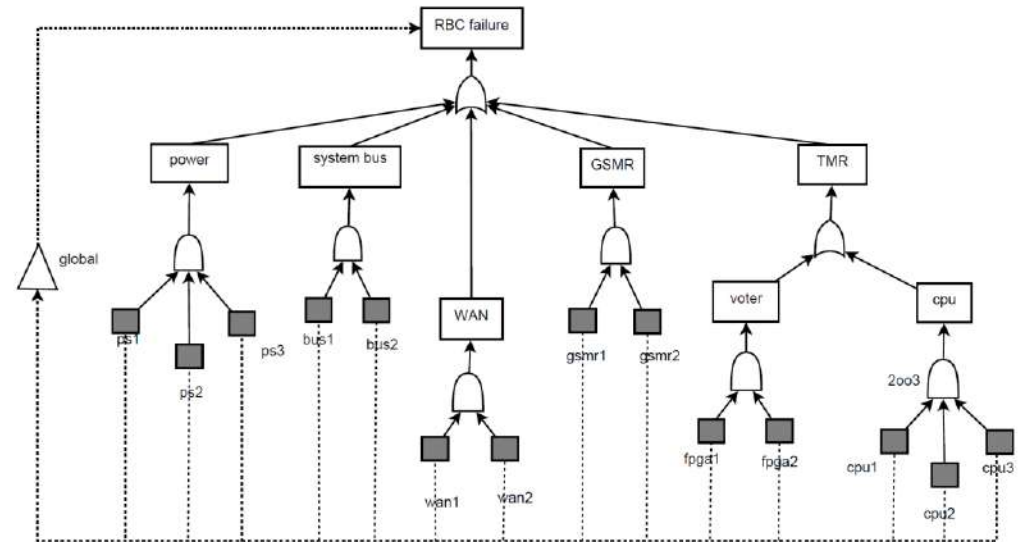


Fig. 7. RFT model of the RBC

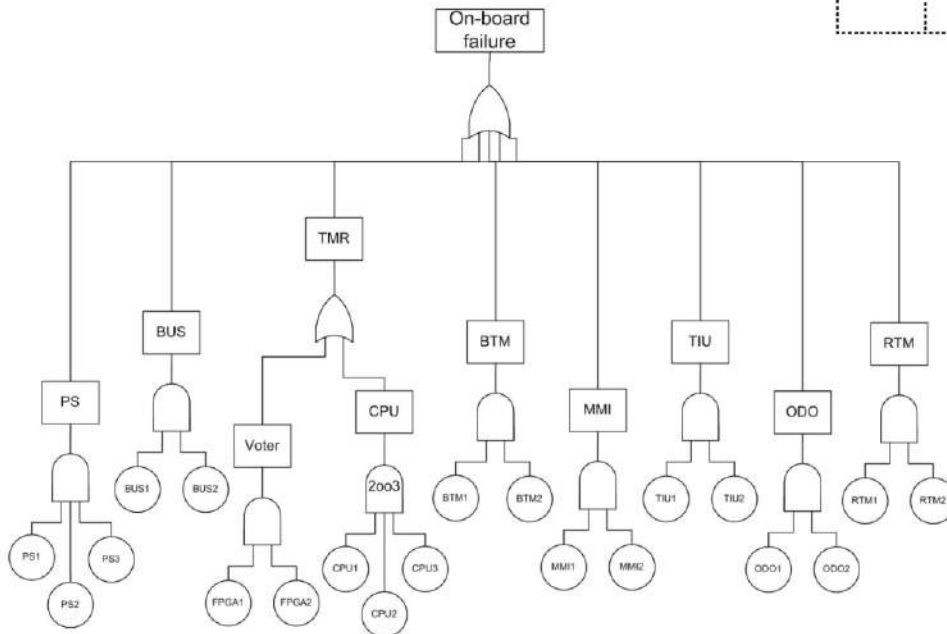


Fig. 6. The FT model for the On-board subsystem

Example: performability

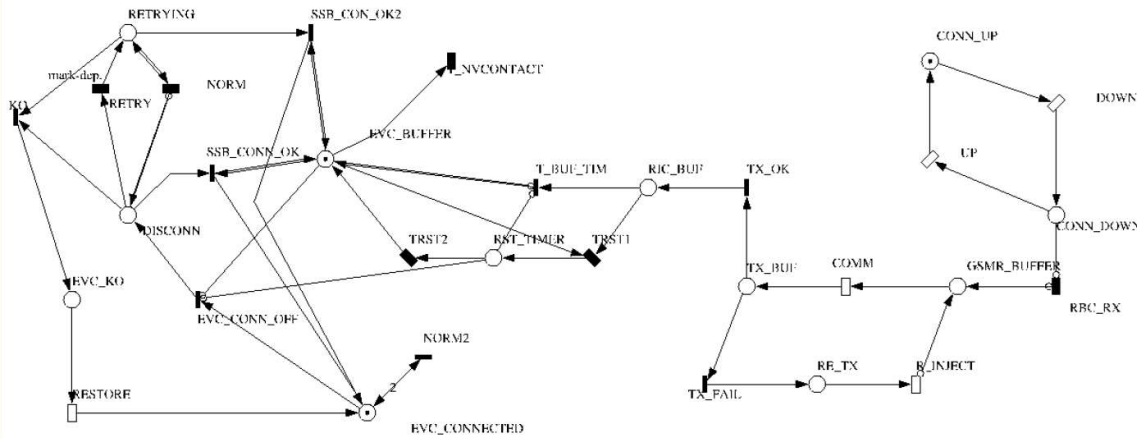


Fig. 9. The structure of the GSPN performability model

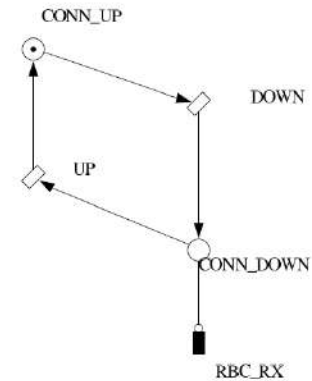


Fig. 10. RBC communication module in vital communication model

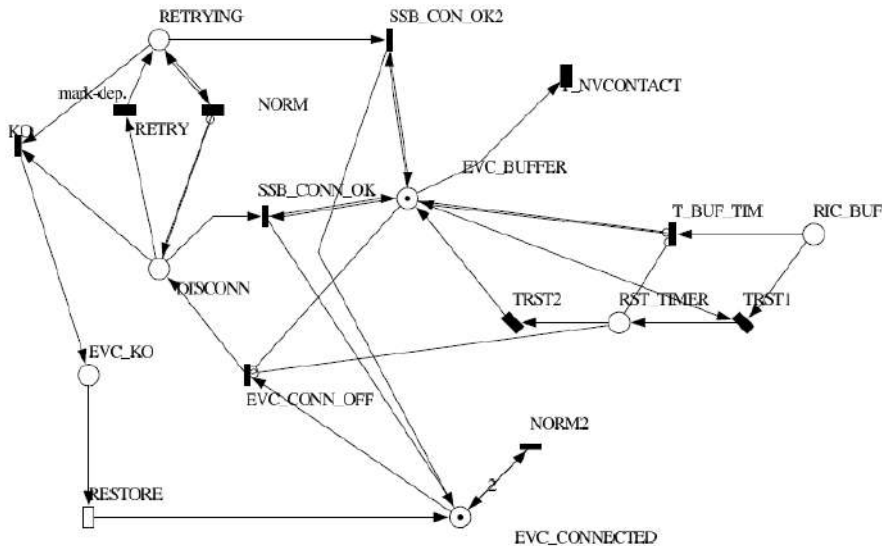


Fig. 12. On-board timing module in vital communication model

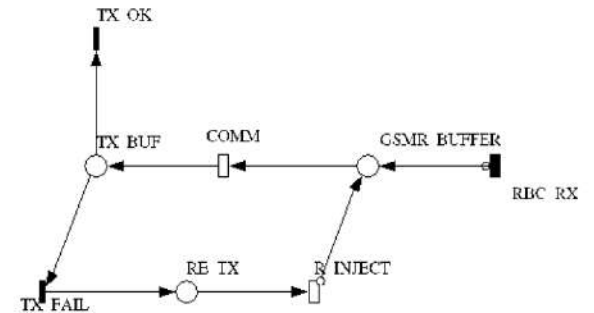


Fig. 11. GSM-R module in vital communication model

So what?

- We analyzed the different submodels for the different aspects of the system
- All submodels showed compliance with specifications and expected system behavior
- **The overall multiformalism system analysis showed that there is a possibility of system failure with no component failure because of combined effects of a legitimate combination of delays of computing and transmissions and a misinterpretation by the control logic**
- Such a problem is a *higher level system effect* not trackable to any of the logs of the components, which show absolutely no error

Doing multiformalism

- What do we need to use multiformalism modeling?
 - Proper model composition tools
 - Proper model analysis strategies
 - Proper model analysis tools
 - A novel theoretical approach

Tools to define multi-formalism models

The screenshot displays the DrawNet software interface. The main window shows a dependency model diagram with nodes like I/O(z), DI(z), Or5, INPUT(z), 2:3(2), IN(y), Or4, TRIBUS(y), CPU(y), INCPU(y), Or3, SIGN(y), DO(y), Or2, SUB(y), 2:3, PS(x), And1, PS, VOTER, CHAN, Or1, and TE. A Property Page window is open, showing details for a ReplicatorEvent named INPUT(z) with visibility false and parameters [z,C2]. A Solver Execution window is also open, displaying XML code for the model.

Property Page

Property	Value
type	ReplicatorEvent
name	INPUT(z)
visibility	false
PList	
Parameters	
PredSWN	[z,C2]
DeclaredParam...	
Label	

Solver Execution

```

<?xml version='1.0' encoding='UTF-8' standalone='no?'>
<PFT name='PFT0' visibility='false' Title=''>
<TopEvent name='TE' visibility='false'/>
<BasicEvent name='VOTER' visibility='false' Distribution='>
<Event name='CHAN' visibility='false' Parameters='' Label='>
<Event name='PS' visibility='false' Parameters='' Label='>
<Or name='Or1' visibility='false'/>
<Arc name='Arc0' visibility='false' from='CHAN' to='Or1'/>
<Arc name='Arc1' visibility='false' from='Or1' to='TE'/>
<Arc name='Arc6' visibility='false' from='PS' to='Or1'/>
<Arc name='Arc7' visibility='false' from='VOTER' to='Or1'/>
<G2of3 name='2:3' visibility='false'/>
<And name='And1' visibility='false'/>
<Arc name='Arc8' visibility='false' from='2:3' to='CHAN'/>
<Arc name='Arc9' visibility='false' from='And1' to='PS'/>
<ReplicatorEvent name='SUB(y)' visibility='false' PList='>
<BasicReplicatorEvent name='PS(x)' visibility='false' PList='>
<Or name='Or2' visibility='false'/>

```

DrawNet++: a Flexible Framework for Building Dependency Models

G. Franceschin, M. Gerardo, M. Iacono, V. Viminini, C. Barnocelli *

1. DrawNet++ quick overview and context

The DrawNet++ project addresses the computational construction of dependency models [5, 6]. Its main goal is to provide a GUI to any graph-based formalism, in support of the design process of dependency models, according to concepts inspired by object-oriented (OO) [7] and finally focused on the different classes of analysis formalisms.

Such features enhance the rapid prototyping and the reuse of sub-models. The formalism adopted to implement the methodology may differ by allowing to study both composition problems related to the usage of a single operation formalism and interoperability issues within a multi-formalism environment. A prototype architecture has been implemented to structure a few graph-manipulation tools and expressive performance results. By name, Parametric Fault Tree (PFT) [8] and Stochastic Well-Formed Nets (SWN) [9] are supported. As XML descriptions of the user-defined models to be generated by the tool and XML styles are used to structure the XML representation into a schema suitable for the specific analysis tool. We use the DrawNet++ framework to build computational SWN models and models of highly redundant systems by creating PFT with SWN [12]. Since state space analysis tools are needed to represent dependencies between components, repair, transient states, etc., we model them by SWN sub-models and compose such sub-models blocks with a SWN model of the system obtained by automatically translating a PFT with SWN [12]. In this context we use the DrawNet++ to facilitate and reuse composed models by PFT and SWN Models to produce the system representation of the entire model and to present the results compactly by the references.



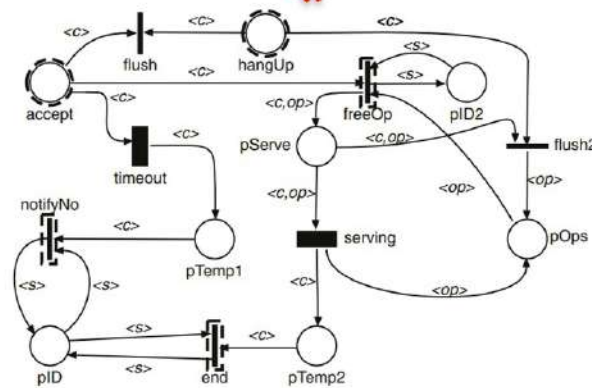
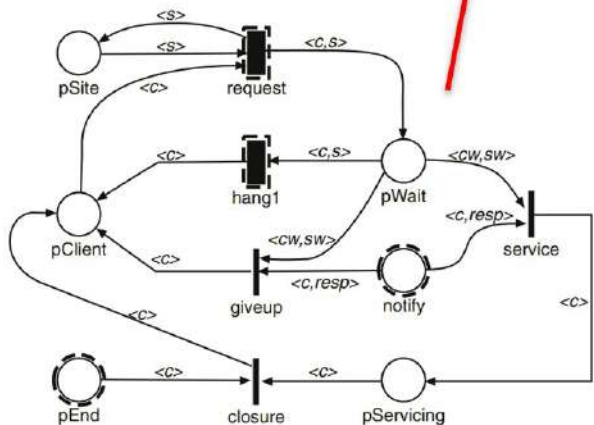
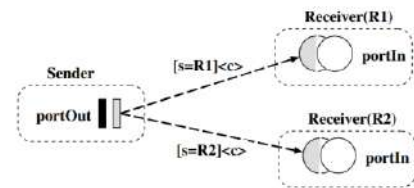
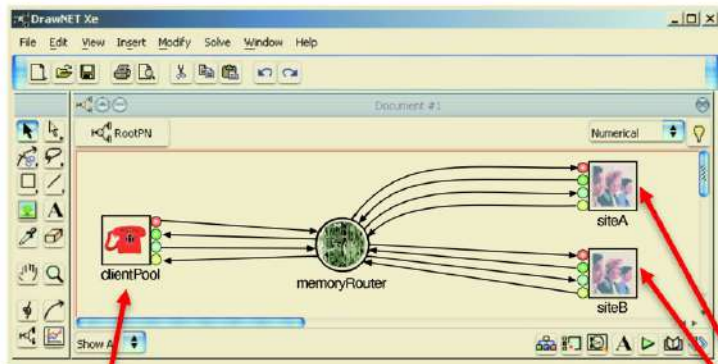
Figure 1. A PFT example.

References

- [1] A. Barba, G. Franceschin, L. Rinaldi, and G. Di. Dependability Assessment of an Industrial Programmable Logic Controller via Parametric Fault Tree and High-Level Petri Nets. Proc. of PFT'01, Las Vegas, September 2001.
- [2] G. Chula, C. Dardano, G. Franceschin, and S. Haddad. Stochastic well-formed nets for concrete model checking applications. IJFST, 42(14):149-160, 1993.
- [3] G. Franceschin, M. Gerardo, M. Iacono, V. Viminini, and C. Barnocelli. DrawNet++: a flexible framework for building dependency models and simulation of complex systems. Proc. of EWASIS'02, London, UK, April 2002.
- [4] M. Gerardo, A. Viminini. Parametric Petri Nets for Performance. Proc. of SNTD'00, Trento, May 2000.

*This work is partially supported by the MUR Project "SINTE".

Multi-solution by orchestration



Compositional Modeling of Complex Systems: Contact Center Scenarios in OsMoSys

Cristiana Franceschini¹, Marco Tribuzzi², Marco Lucano³, Stefano Marrone⁴, Nicola Mazzecca³, and Valeria Vittofini¹

¹ Dipartimento di Informatica, Università del Piemonte Orientale, Piazza Amicoeoli 5, I-10100, Alessandria, Italy
giulianaf@unipao.it

² Dipartimento di Informatica, Università di Torino, Corso Svizzera 155/B, 10149 Torino, Italy
marco@di.unito.it

³ Dipartimento di Ingegneria dell'Informazione, Seconda Università di Napoli, Via Roma 29, 81031 Arzano, Italy

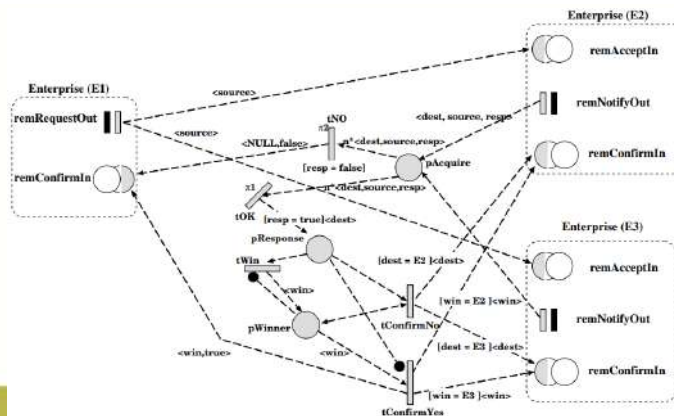
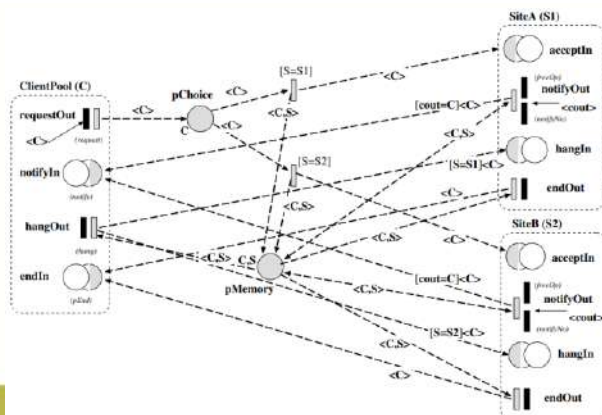
⁴ Dipartimento di Informatica e Sistemistica, Università degli Studi di Napoli "Federico II", Via Claudio 21, 80125 Napoli, Italy
vittorin@unina.it

Abstract. In this paper we present the application of a compositional modeling methodology to the engineering of Stochastic Well Formed net (SWN) models of a contact center. The modeling methodology is based on the definition of proper operators to connect submodels and it is supported by the OsMoSys modeling framework. The paper describes the implementation of a library of reusable SWN submodels of the contact center components and the definition of proper SWN connectors to easily develop models of different configuration of the system. We also describe the solving process of the composed models and its integration in the OsMoSys framework. Moreover, we discuss the advantages that this approach, based on the definition of closed and instances of sub-models, can provide to the application of SWN to complex case studies.

1 Introduction

Compositionality is an effective mean to cope with the complexity of real system modeling and analysis. It can also be exploited to set up a framework where designers, who may not be experts in formal modeling languages, compose models of complex systems by just choosing building blocks from a library and connecting them. The building blocks could hide complex models built by experts.

In the context of Petri nets (PN) a number of proposals have appeared in the literature to add model (de)composition features to a formalism that is not compositional in its basic definition (e.g. [1,3,10,11]). In particular, in [11, 14] PN are used to describe the behavior of object classes and communication mechanisms are defined to allow the object models to inter-operate. The approach proposed in [7] adapts this model composition paradigm to Stochastic



“Real” multi-formalism models

- Multiformalism potential is not limited to a joint analysis of heterogeneous submodels
- In the Mobius approach a superformalism towards which all formalisms are translated allow the generation of a single analysis model
- In the SIMTHESys approach *elements from different formalisms may actually natively interact* by specifying their elementary interactions and complex state change conditions, to account for actual behaviors of a system

Element Based Semantics in Multi Formalism Performance Models

M. Baroni¹, M. Colabato²

¹ Dipartimento di Ingegneria e Informatica, Università degli Studi di Napoli, Capua, Italy
² Dipartimento di Informatica, Politecnico di Milano, Milano, Italy
marco.colabato@polimi.it

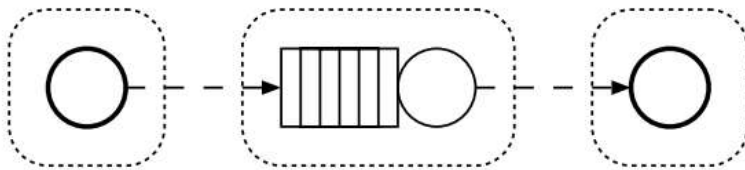
Abstract. The design and the implementation of modern computer-based systems are made by complexity that calls for the use of models for the verification of non functional requirements since the beginning of their design cycle, such systems are however too complex to be modeled directly in a single mathematical formalism: the Object-Oriented Petri Nets, the Petri Nets, the Petri Nets (Downward) and the Petri Nets (Upward) modeling, and the modeling of heterogeneous formalisms and formalisms in systems. A novel approach to multi-formalism compositional modeling, that is based on the possibility of their specific: the definition of the elements of a formal modeling language to be used in the modeling. This is obtained by the application of compositional modeling techniques to the description of models, together with the concept of behavior as a single system formalism description and system synthesis. In this paper the main concepts of the SIMTHESys approach are presented, together with a running example of four SIMTHESys steps with performance evaluation of multi-formalism models.

1. INTRODUCTION

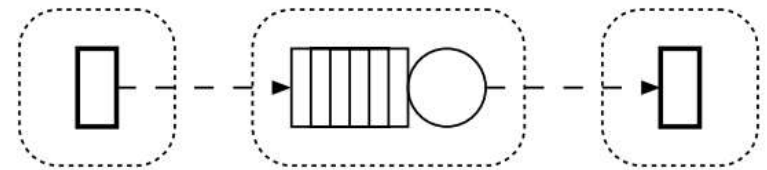
In many years computer systems like Computer Networks, Multi-User Applications, Web Services, Critical Infrastructures, Controllers have reached a complexity level that suggest the use of modeling techniques during their design phase. The main systems are modeled in languages like the Querying Networks or the Petri Nets, Component Based notations [1] while the compositionality of large systems is collected in modeling sets systems. At the same time, substitution of a complex model into components suggest the possibility of describing sets of the sub-systems using a different language. Multi-Formalism modeling techniques allow the definition of the components describing the complex model to be specified using the same appropriate formalism. One of the problems of multi-formalism is that it requires the development of software solution components that are not specific to the particular set of interacting formalisms considered. Usually, the definition of a new formalism involves an existing one, requires a large amount of software of the system that is not usually straightforward and can result in an evolutionary design and more development effort with respect to the benefits. However, a new specific combination of formalisms is very difficult to translate into other people. Thus, a solution component also to be developed only for a very specific problem. In this work, we propose a way of encapsulating the evolution

“Real” multi-formalism models

- The main idea was finding an object-oriented way of describing the interactions between modelling primitives
- The abstraction offered by OOP could immediately define an interconnection semantic between primitives of different formalisms



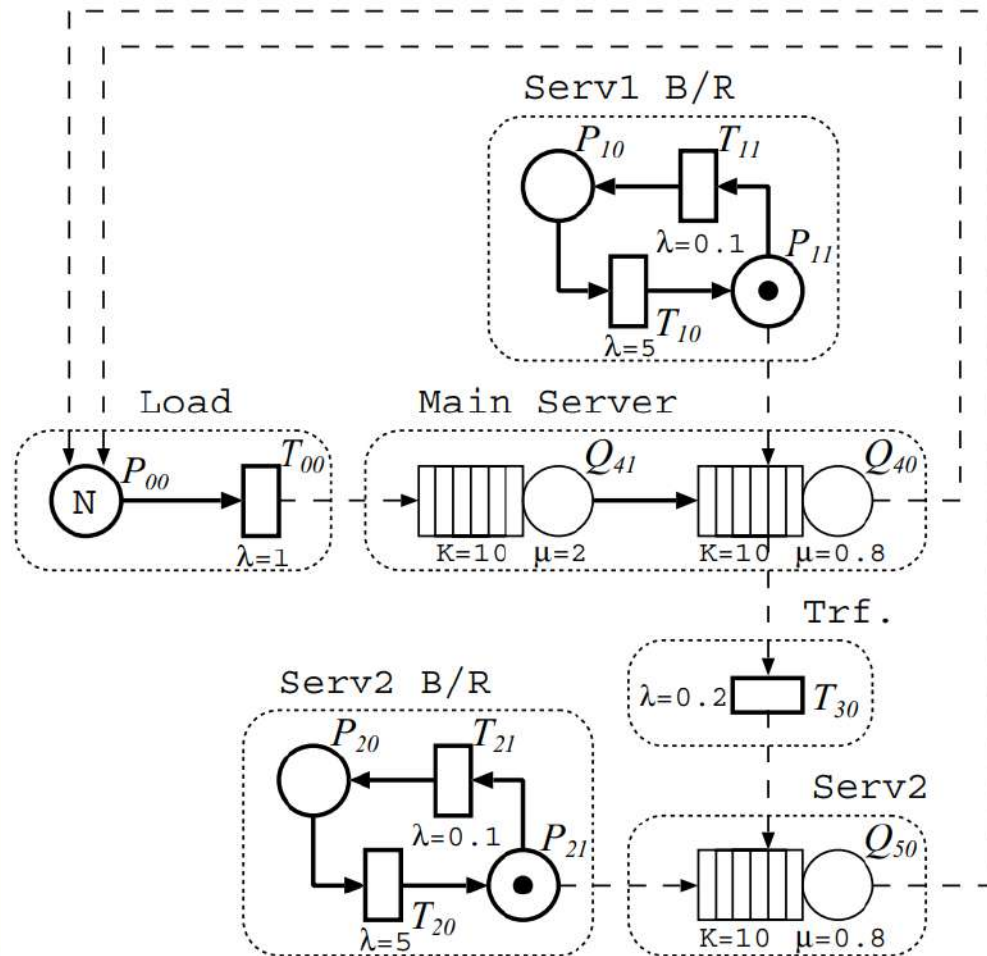
a)



b)

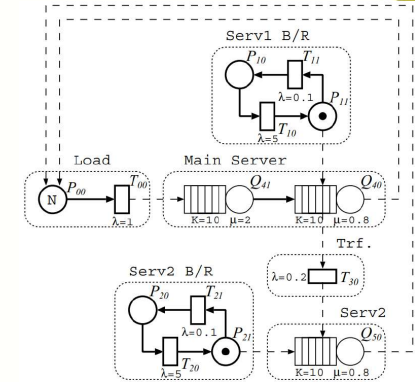
“Real” multi-formalism models

- A closed system, with servers that can break and repair, together with the possibility of transferring a job to a secondary system in case of long waits

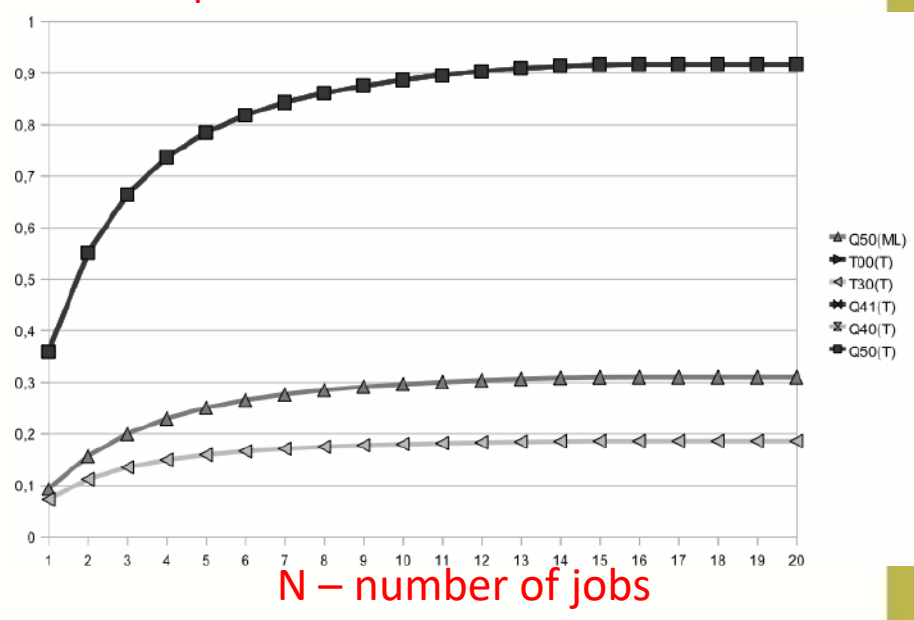
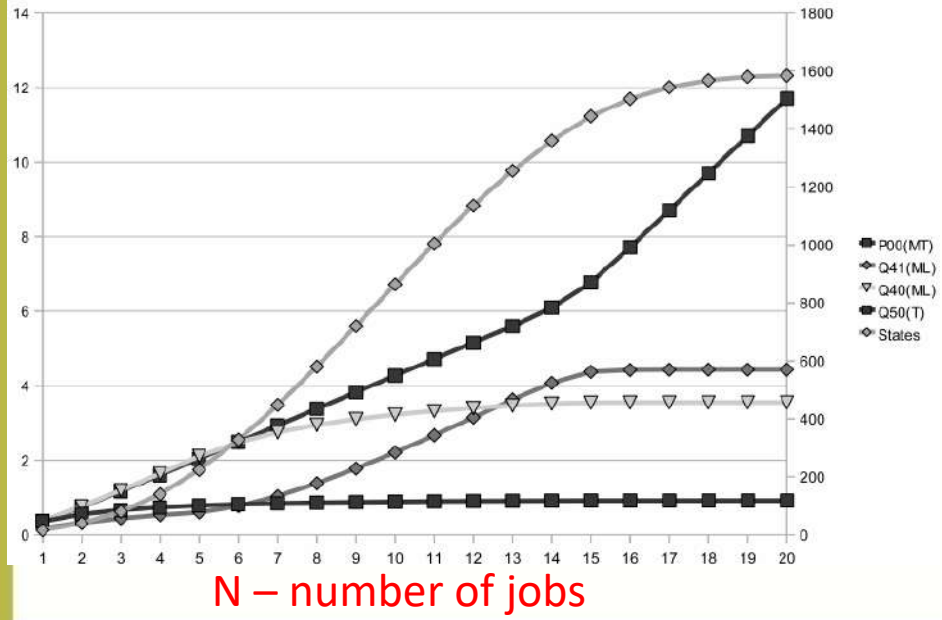


“Real” multi-formalism models

- The proposed rules allow either to produce a Continuous Time Markov Chain for numerical analysis, or to compute a solution using discrete events simulation



Transient evolution of the "states" of the primitives.



Other interesting cases

- Specifying complex verification properties
 - Petri nets to specify behaviors, Fault Trees to evaluate conditions, automata to assess properties
- Modeling the effects of software rejuvenation policies
 - Combining a traditional modelling language with a *Domain Specific Language* properly defined to represent alternate software rejuvenation policies
- Hybrid systems
 - Modeling systems with continuous time and discrete time components

Exploiting multimodal models for testing and performance evaluation in SIMTHESys

F. Baccardo
Department of Information Systems and Computing
University of Calabria
I-87030 Arcavacata (CS)
Italy
fbaccardo@unical.it

M. Coluccia
Department of Computer Science
University of Calabria
I-87030 Arcavacata (CS)
Italy
mcoluccia@unical.it

M. Iacono
Department of Information Systems and Computing
University of Calabria
I-87030 Arcavacata (CS)
Italy
miacono@unical.it

ABSTRACT

This paper shows how multimodal models can be exploited to model hybrid systems exhibiting performance and reliability characteristics. The following combination of multimodal modeling is used: Petri nets to specify behaviors, Fault Trees to evaluate conditions, automata to assess properties. The paper shows how multimodal models can be exploited to model hybrid systems exhibiting performance and reliability characteristics. The following combination of multimodal modeling is used: Petri nets to specify behaviors, Fault Trees to evaluate conditions, automata to assess properties.

1. Introduction and related works

The paper shows how multimodal models can be exploited to model hybrid systems exhibiting performance and reliability characteristics. The following combination of multimodal modeling is used: Petri nets to specify behaviors, Fault Trees to evaluate conditions, automata to assess properties.

KEYWORDS

Hybrid systems, multimodal models, performance evaluation, reliability evaluation.

1. INTRODUCTION

Hybrid systems are systems that exhibit both continuous and discrete behaviors. They are characterized by the presence of both continuous and discrete components. This paper shows how multimodal models can be exploited to model hybrid systems exhibiting performance and reliability characteristics. The following combination of multimodal modeling is used: Petri nets to specify behaviors, Fault Trees to evaluate conditions, automata to assess properties.

RELATED WORKS

The paper shows how multimodal models can be exploited to model hybrid systems exhibiting performance and reliability characteristics. The following combination of multimodal modeling is used: Petri nets to specify behaviors, Fault Trees to evaluate conditions, automata to assess properties.

REFERENCES

1. Introduction and related works

The paper shows how multimodal models can be exploited to model hybrid systems exhibiting performance and reliability characteristics. The following combination of multimodal modeling is used: Petri nets to specify behaviors, Fault Trees to evaluate conditions, automata to assess properties.

Modeling Hybrid Systems in SIMTHESys

Fabrizio Baccardo¹, Marco Coluccia² and Mauro Iacono³

¹Department of Information Systems and Computing, University of Calabria, I-87030 Arcavacata (CS), Italy
²Department of Computer Science, University of Calabria, I-87030 Arcavacata (CS), Italy
³Department of Information Systems and Computing, University of Calabria, I-87030 Arcavacata (CS), Italy

ABSTRACT

This paper shows how multimodal models can be exploited to model hybrid systems exhibiting performance and reliability characteristics. The following combination of multimodal modeling is used: Petri nets to specify behaviors, Fault Trees to evaluate conditions, automata to assess properties.

1. Introduction and related works

The paper shows how multimodal models can be exploited to model hybrid systems exhibiting performance and reliability characteristics. The following combination of multimodal modeling is used: Petri nets to specify behaviors, Fault Trees to evaluate conditions, automata to assess properties.

REFERENCES

1. Introduction and related works

The paper shows how multimodal models can be exploited to model hybrid systems exhibiting performance and reliability characteristics. The following combination of multimodal modeling is used: Petri nets to specify behaviors, Fault Trees to evaluate conditions, automata to assess properties.

KEYWORDS

Hybrid systems, multimodal models, performance evaluation, reliability evaluation.

Rethinking PN/QN connections

- We defined non-traditional PN/QN connections
- We started considering possible applicative scenarios of the connections, aiming at making models "readable"

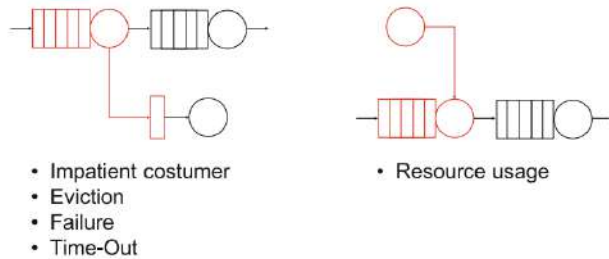


Fig. 2. Advanced connection types: queue to transition and place to queue.

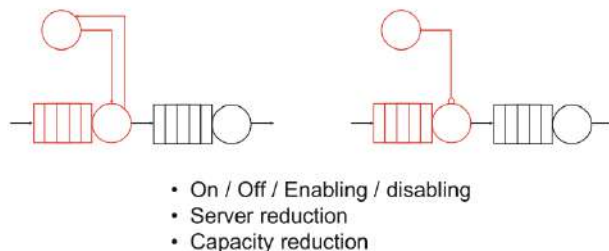


Fig. 3. Advanced connection types: test and inhibitor arcs.

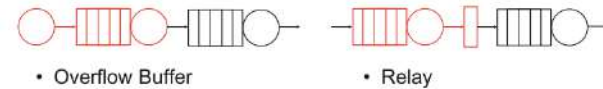


Fig. 4. Advanced connection types: exclusive connection from place to queue and from queue to transition.

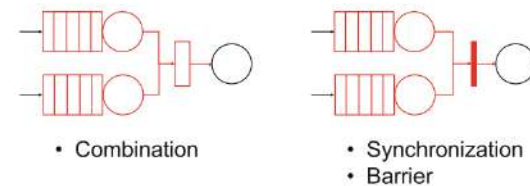


Fig. 5. Advanced connection types: from several queues directly to a timed or immediate transition.

Places, Transitions and Queues: New Proposals for Petri-Net Semantics

Roberto Colaneri¹ and Klaus Inacker^{2,3}

¹ Politecnico di Milano, via Pavesi 15, 20133 Milano, Italy
 e-mail: roberto.colaneri@polimi.it

² Universität Bayreuth, Postfach 1012, 90402 Bayreuth, Germany
 e-mail: klaus.inacker@uni-bayreuth.de

³ Universität Bayreuth, Postfach 1012, 90402 Bayreuth, Germany
 e-mail: klaus.inacker@uni-bayreuth.de

Abstract. In this paper we discuss some novel connections between the semantics of Petri-net models and the semantics of Petri-net models. These connections are to be used to model the semantics of Petri-net models. The connections are to be used to model the semantics of Petri-net models. The connections are to be used to model the semantics of Petri-net models.

Keywords. Petri nets, Coloured Petri nets, Semantics, Petri-net models, Petri-net models.

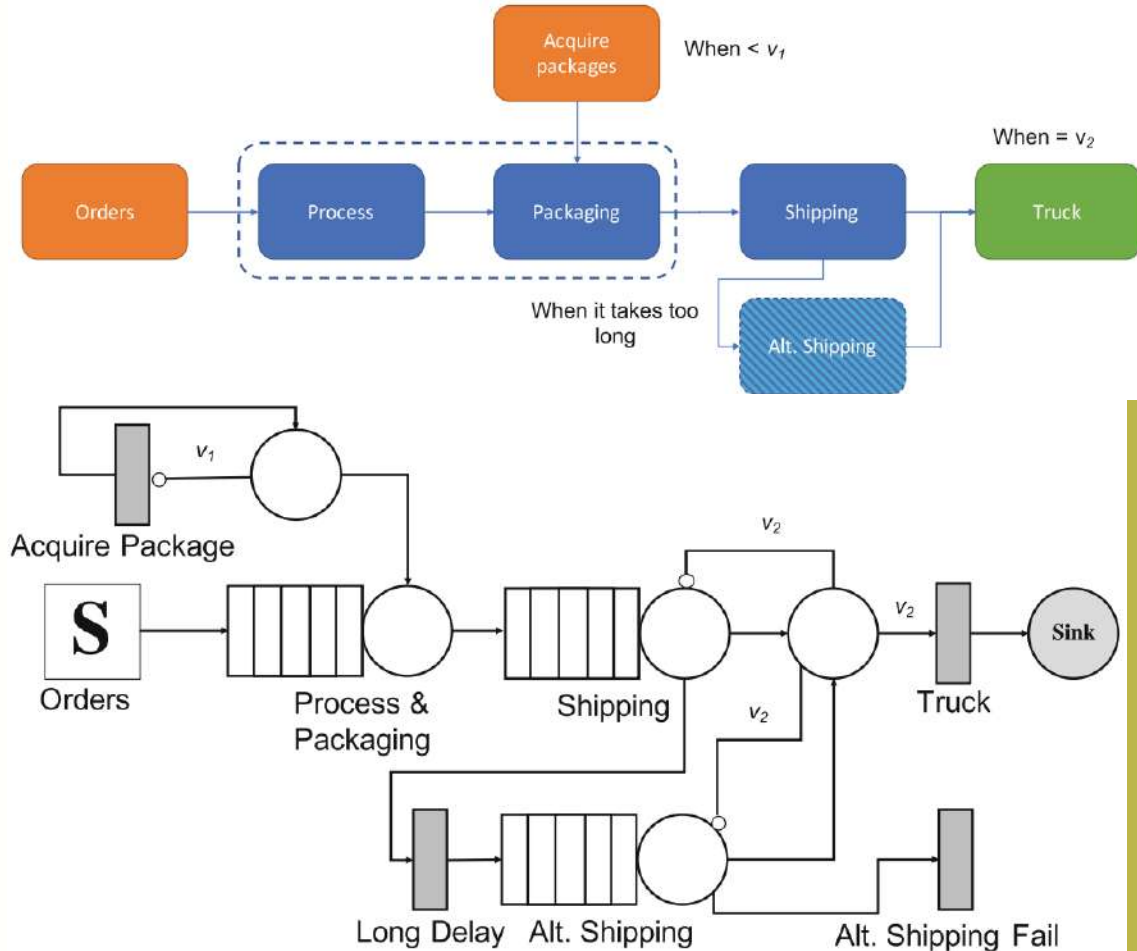
1 Introduction

In this paper we discuss some novel connections between the semantics of Petri-net models and the semantics of Petri-net models. These connections are to be used to model the semantics of Petri-net models. The connections are to be used to model the semantics of Petri-net models. The connections are to be used to model the semantics of Petri-net models.

© 2013 IEEE. Personal use of this paper is permitted. All rights reserved. This paper is intended only for individual users. All rights are reserved. No part of this paper may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or by any information storage and retrieval system, without permission in writing from IEEE. For more information on this copyright notice please go to the IEEE website at www.ieee.org.

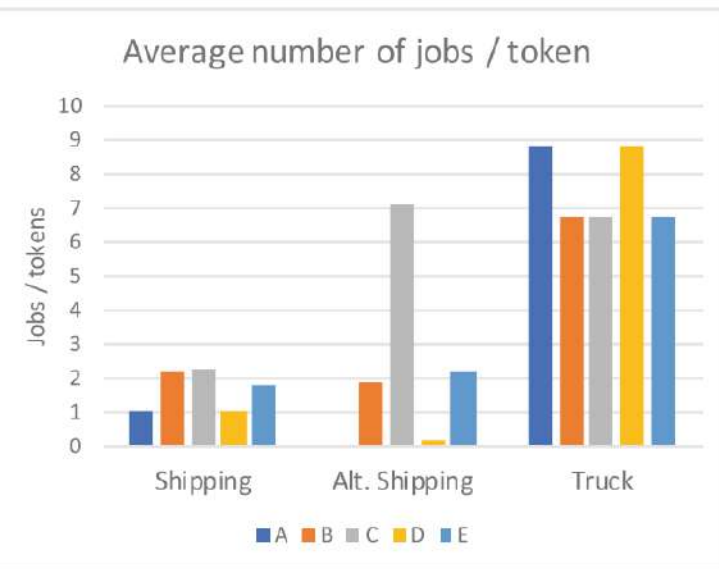
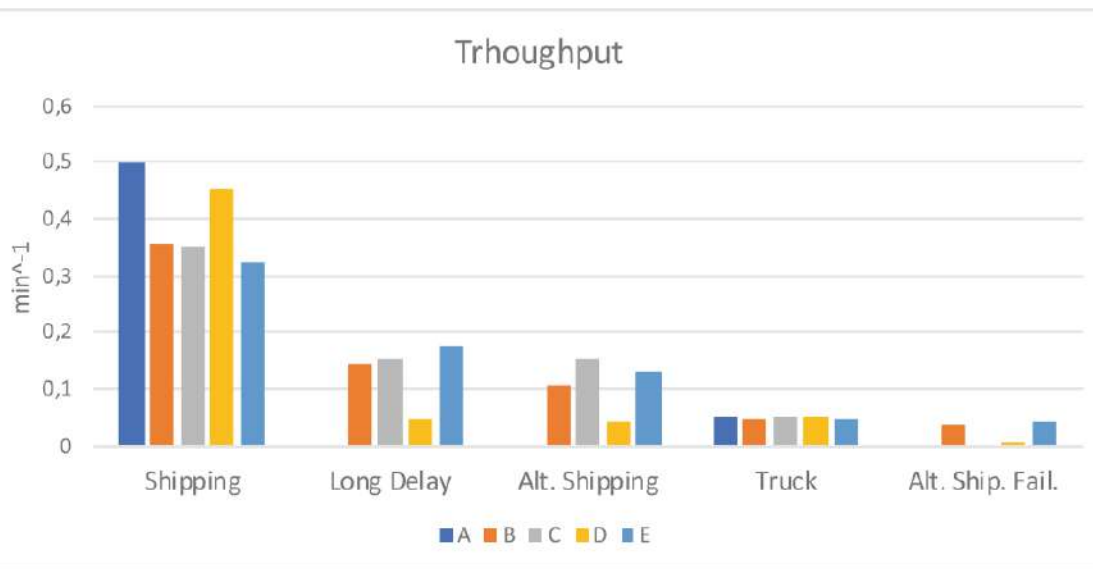
Rethinking PN/QN connections

- We tried to test different semantics, in a simple goods delivery model
- These semantic focused on differences between jobs in service and waiting in the queue



Rethinking PN/QN connections

- We showed that the different semantics have a strong impact on the performance measures that can be computed on the models:
 - they cannot be simply ignored, or left unspecified!



Questions?

The ERTMS/ETCS running example models are from Francesco Flammini, Stefano Marrone, Mauro Iacono, Nicola Mazzocca, Valeria Vittorini, A Multiformalism Modular Approach to ERTMS/ETCS Failure Modelling. International Journal of Reliability, Quality and Safety Engineering, vol. 21, num. 1, pp. 1450001-1-1450001-29, World Scientific , ISSN: 0218-5393, DOI: 10.1142/S0218539314500016

My gratitude goes to my friend prof. Marco Gribaudo for 25 years of work together and for providing part of the materials used in this presentation



● Università
● degli Studi
della Campania
Luigi Vanvitelli

Prof. Mauro Iacono

Professore associato in Sistemi di Elaborazione delle Informazioni
mauro.iacono@unicampania.it