PMR 5237
Modelagem e Design de Sistemas Discretos em Redes de Petri

Aula 12: Perspectivas de pesquisa em RdP

Prof. José Reinaldo Silva
reinaldo@usp.br

Plano de Aulas

Aula	Tema	Data
Aula7	Propriedades das redes P/T e Coloridas	9/11
Aula8	Análise de Invariantes	16/11
Aula9	Técnicas de modelagem	23/11
Aula10	Métodos de Design orientados a estados	30/11
Aula11	Métodos de Design orientados a eventos	07/12
Aula12	Perspectivas de pesquisa em modelagem de sistemas discretos com RdP	14/12

	PMR3510	PMR3410	PMR5237
9/12	Entrega do EP (meia-noite)		Milestone do artigo (7/12)
14/12		Entrega do Pitch (video 3min) da Startup	Milestone do artigo (14/12)
19/12	Entrega da monografia		
21/12		Entrega da documentação da proposta de Startup	Artigo Final (21/12)

Journal of Educational Psychology
2016, Vol. 108, No. 5, 740–750

© 2016 American Psychological Association
0022-0663/16/$12.00 http://dx.doi.org/10.1037/edu0000086

Peer Play as a Context for Identifying Profiles of Children and Examining Rates of Growth in Academic Readiness for Children Enrolled in Head Start

Elizabeth R. Bell, Daryl B. Greenfield, Rebecca J. Bulotsky-Shearer, and Tracy M. Carter
University of Miami

Research has shown that early interventions are most successful when they have a comprehensive focus that is individualized to children's needs. The present study employed a person-centered approach to identify profiles, or subgroups, of children displaying early patterns of peer play behaviors in an ethnically and linguistically diverse Head Start program, and examined the academic trajectories of these children during one school year. Four profile groups were identified, and analyses revealed that these profiles were invariant across ethnicity and dual language learner status. Most children were represented in a group who engaged in behaviors that facilitated peer interactions. These children had the highest academic skills across the preschool year. Interestingly, children in a profile characterized by a combination of play interaction skills and play disruption had the second highest academic skills throughout the year compared with children in a profile characterized by below-average play interaction skills but low disruptive behavior during play. A small number of children were represented in a profile characterized by low interactive, disconnected, and high disruptive behavior with peers and had the lowest academic skills throughout the year. The mean differences in academic skills across profiles of peer play behaviors remained the same across the year. These findings have implications for future research and educational practice surrounding the role of peer play in the Head Start classroom.

Keywords: Head Start, peer play behaviors, school readiness, latent profile analysis, whole-child approach

Children living in poverty are at increased risk for exposure to environmental stressors and limited access to adequate resources (e.g., family stress, lack of desirable housing, exposure to community violence; G. J. Duncan, Brooks-Gunn, & Klebanov, 1994). Experiencing these multiple stressors places children at additional risk for difficulties adjusting to formal schooling, often leading to poor academic achievement, particularly compared with their middle- and high-income peers (O. Lee & Buxton, 2010). Unfortunately, evidence suggests that the achievement gap between low-income and high-income students is continually increasing (Reardon, 2013). Research identifying and promoting emergent competencies, how they vary among children, and how they are associated with academic learning is needed to inform how to best protect these vulnerable children from experiencing difficulties upon entry into school (Barbarin, 2007; Kagan, Moore, & Bredekamp, 1995).

Early intervention programs, such as Head Start, have the potential to alleviate the risks of poverty associated with poor school adjustment and achievement (V. E. Lee & Burkam, 2002; Shonkoff & Phillips, 2000). Research has shown that such interventions are most successful when they are comprehensive and flexible to meet each individual child's specific needs (Ramey & Ramey, 1998). Head Start is the largest federally funded early childhood program in the United States serving predominantly low-income children. Since its inception, Head Start's comprehensive intervention approach has focused on promoting the development of the whole child (Zigler & Bishop-Josef, 2006). Head Start provides educational, health, and social services to low-income children and their families with the goal of promoting children's development across multiple domains, including cognitive, social, emotional, and physical. Specifically, Head Start performance standards mandate that classrooms must utilize social interactions to support each child's cognitive and language skills by "using various strategies including experimentation, inquiry, observation, play and exploration" (1304.21 [a] [4] [i]; U.S. Department of Health and Human Services, 2006, p. 70).

Developmental theory and research suggest that the preschool classroom is a naturally occurring context for peer play through which children acquire knowledge and skills (Coplan & Arbeau, 2009; D. G. Singer, Golinkoff, & Hirsh-Pasek, 2006). A growing body of research conducted in Head Start classrooms provides concurrent and longitudinal evidence for the positive associations between behaviors that facilitate peer play and language and mathematics skills in preschool, kindergarten, and third grade (Bulotsky-Shearer, Bell, Romero, & Carter, 2012; Fantuzzo, Sekino, & Cohen, 2004; Hampton & Fantuzzo, 2003; Sekino, 2006). In addition, research has also found that behaviors that interfere with peer play are associated with poorer academic

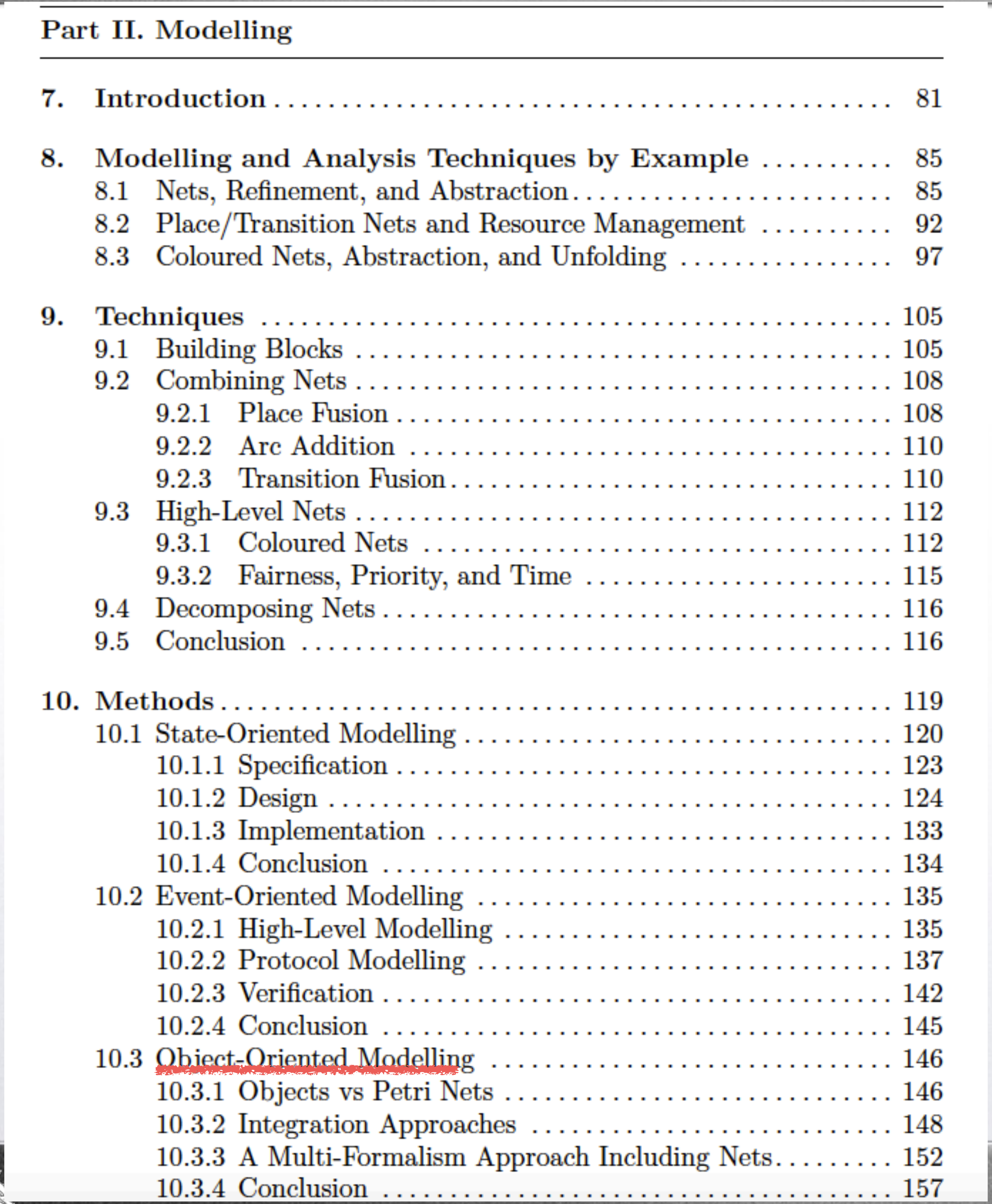
This article was published Online First January 11, 2016.

Elizabeth R. Bell, Daryl B. Greenfield, Rebecca J. Bulotsky-Shearer, and Tracy M. Carter, Department of Psychology, University of Miami.

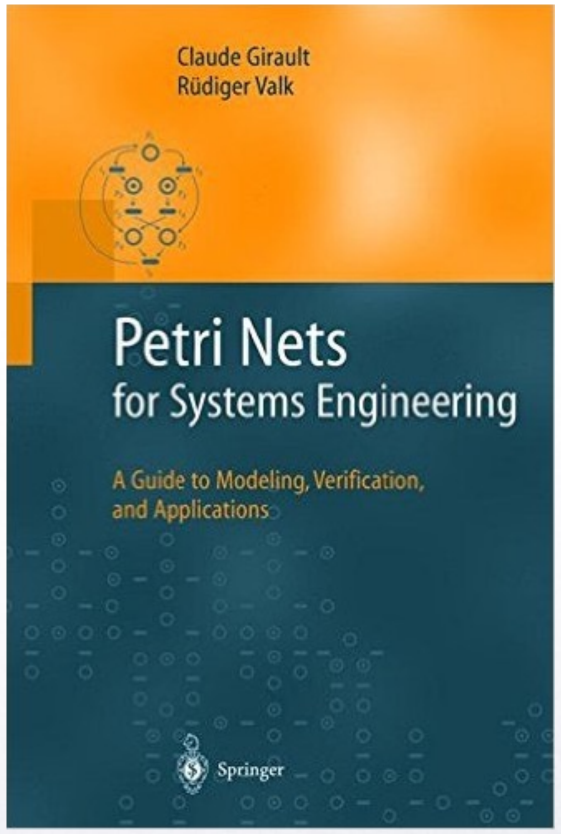
Correspondence concerning this article should be addressed to Elizabeth R. Bell, who is now at Accelerate Learning Inc., 5700 Richmond Avenue, Suite 1025, Houston, TX 77056. E-mail: ebell@acceleratelearning.com

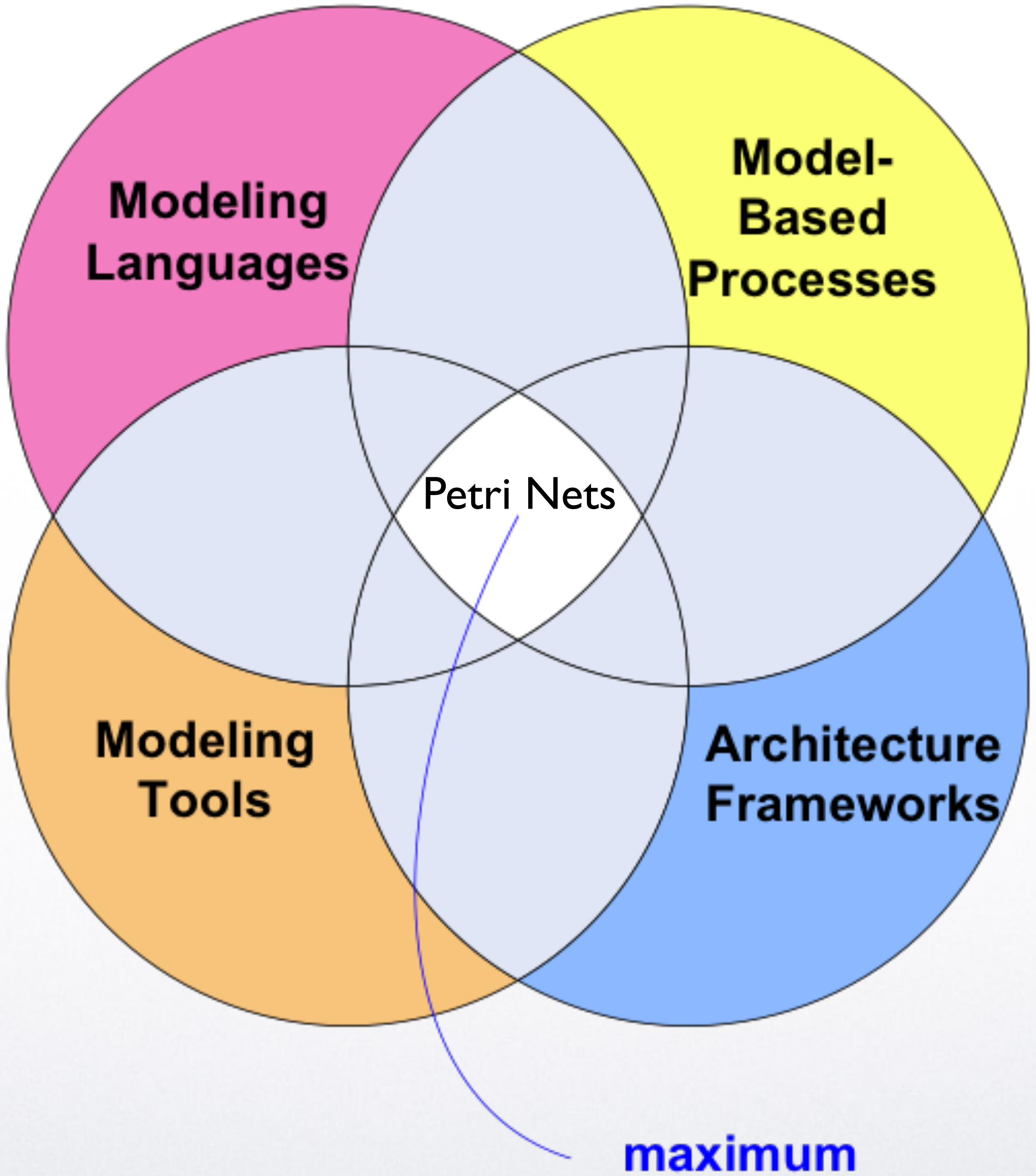
740

Artigo Final: teremos o milestone final nesta aula e reservaremos alguns minutos no final da aula para discussão. O artigo final fica para o dia 21/12.



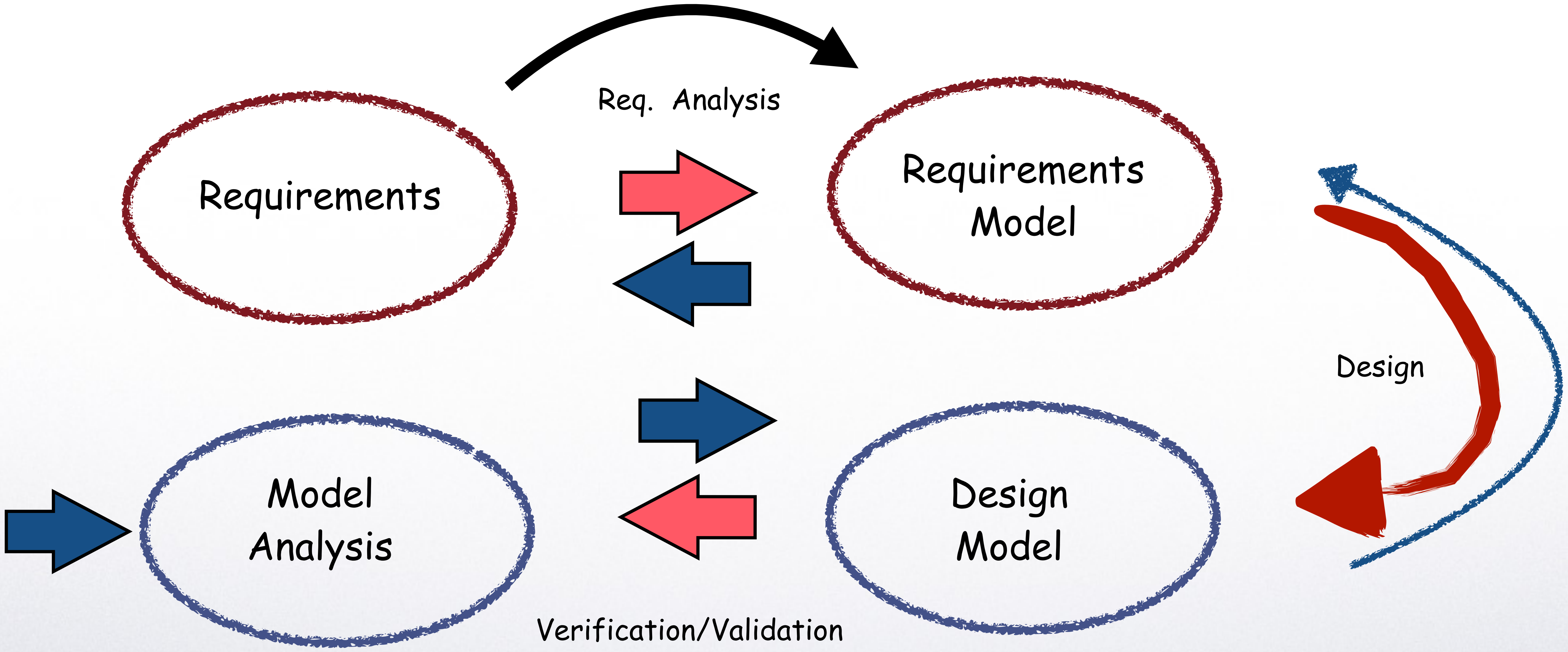
Part II. Modelling

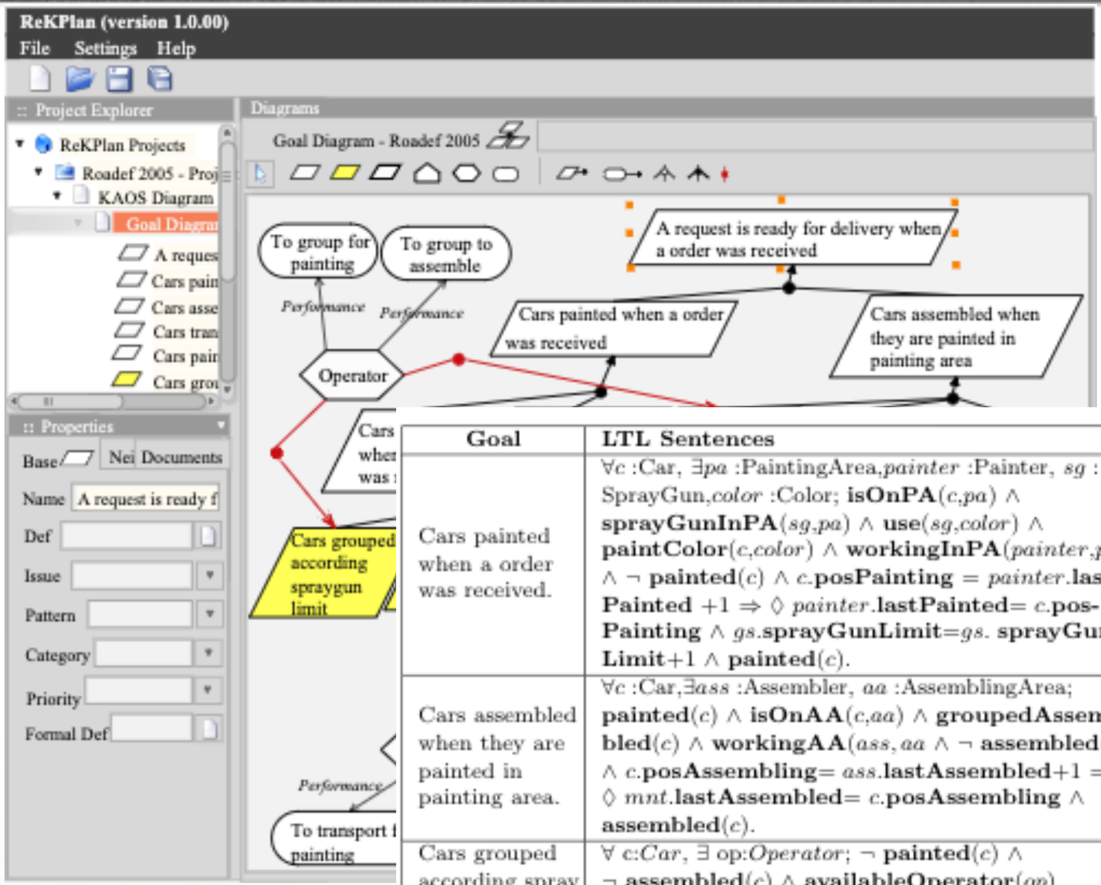




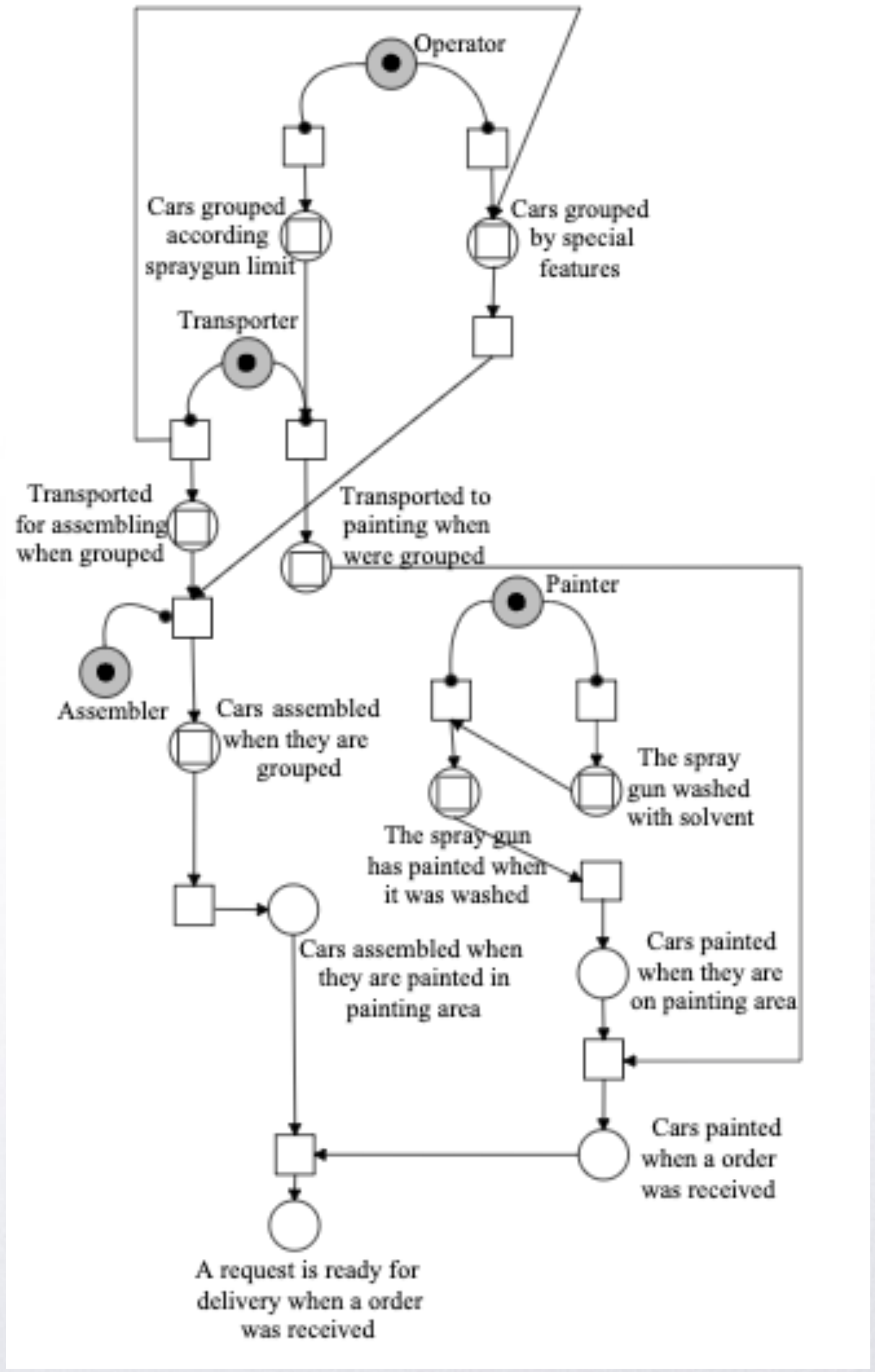
maximum system architecture effectiveness

Use of Petri Nets in Design



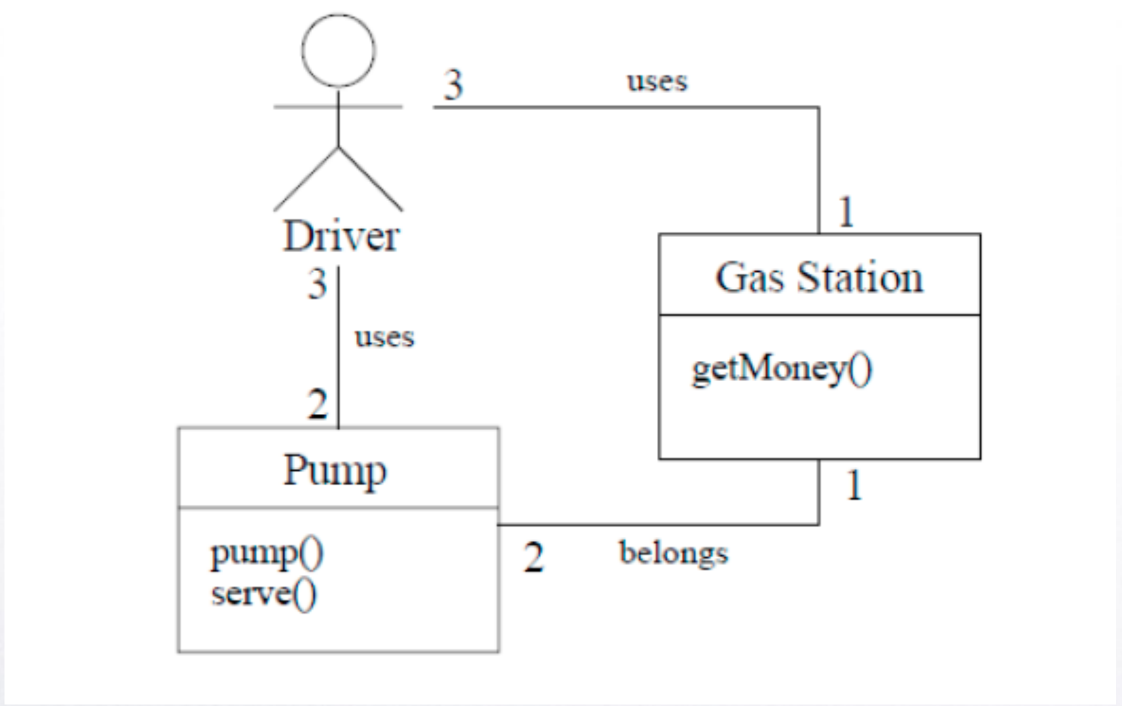


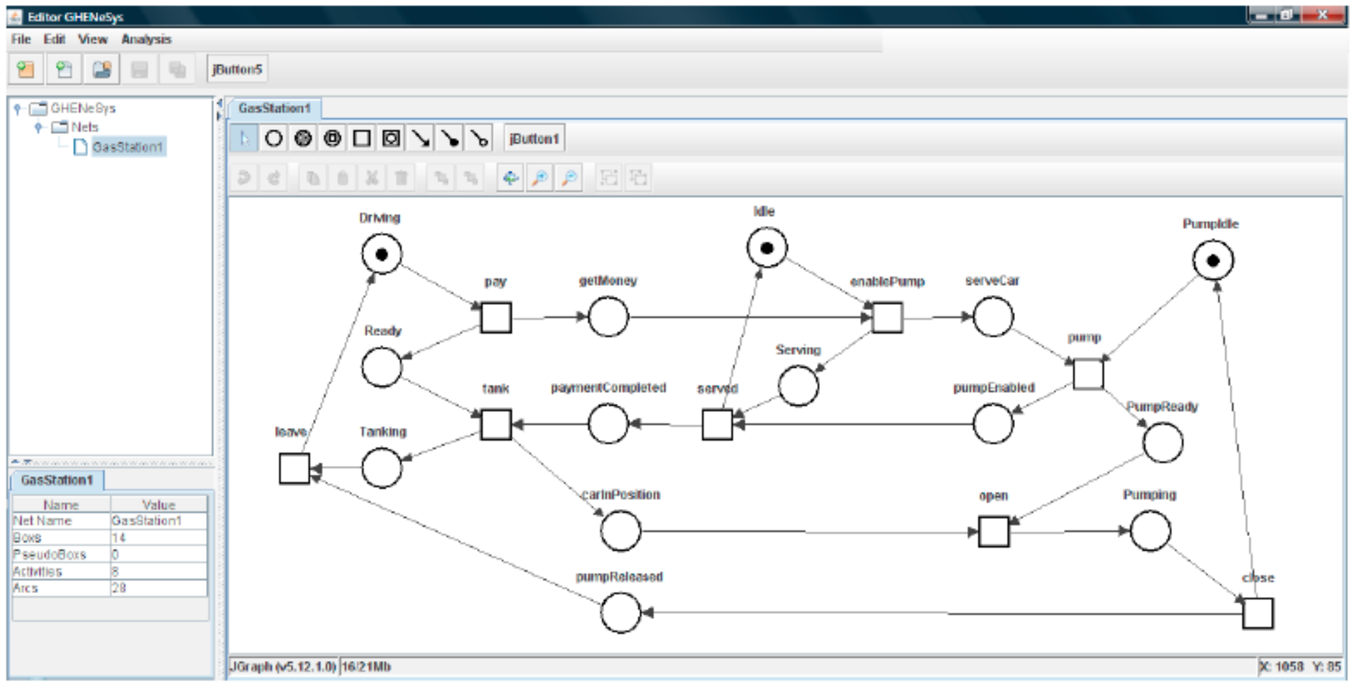
Goal	LTL Sentences
Cars painted when a order was received.	$\forall c:\text{Car}, \exists pa:\text{PaintingArea}, painter:\text{Painter}, sg:\text{SprayGun}, color:\text{Color}; \mathbf{isOnPA}(c,pa) \wedge \mathbf{sprayGunInPA}(sg,pa) \wedge \mathbf{use}(sg,color) \wedge \mathbf{paintColor}(c,color) \wedge \mathbf{workingInPA}(painter,pa) \wedge \neg\, \mathbf{painted}(c) \wedge c.\mathbf{posPainting} = painter.\mathbf{lastPainted} +1 \Rightarrow \Diamond\, painter.\mathbf{lastPainted} = c.\mathbf{posPainting} \wedge gs.\mathbf{sprayGunLimit} = gs.\mathbf{sprayGunLimit}+1 \wedge \mathbf{painted}(c).$
Cars assembled when they are painted in painting area.	$\forall c:\text{Car}, \exists ass:\text{Assembler}, aa:\text{AssemblingArea}; \mathbf{painted}(c) \wedge \mathbf{isOnAA}(c,aa) \wedge \mathbf{groupedAssembled}(c) \wedge \mathbf{workingAA}(ass, aa \wedge \neg\, \mathbf{assembled}(c) \wedge c.\mathbf{posAssembling} = ass.\mathbf{lastAssembled}+1 \Rightarrow \Diamond\, mnt.\mathbf{lastAssembled} = c.\mathbf{posAssembling} \wedge \mathbf{assembled}(c).$
Cars grouped according spray gun limit.	$\forall\, c:Car, \exists\, \text{op}:Operator; \neg\, \mathbf{painted}(c) \wedge \neg\, \mathbf{assembled}(c) \wedge \mathbf{availableOperator}(op) \wedge c.\mathbf{posPainting} = 0 \Rightarrow \Diamond\, \mathbf{groupedPaint}(c).$
Cars grouped by special features.	$\forall c:\text{Car}, \exists\, \text{op}:Operator; \mathbf{painted}(c) \wedge \mathbf{availableOperator}(op) \wedge \neg\, \mathbf{groupedAssembled}(c) \Rightarrow \Diamond\, \mathbf{groupedAssembled}(c).$
Transported to painting when were grouped.	$\forall c:\text{Car}, \exists tra:\text{Transporterr}, \exists pa:\text{PaintingArea}, sg:\text{SprayGun}; \mathbf{groupedPaint}(c) \wedge \mathbf{availableTransporter}\ (tra) \wedge \neg\, \mathbf{painted}(c) \wedge \neg\, \mathbf{isOnPA}(c,pa) \wedge pa.\mathbf{currentPaint} < sg.\mathbf{sprayGunLimit} \Rightarrow \Diamond\, \mathbf{isOnPA}(c,ap) \wedge pa.\mathbf{currentPaint} = pa.\mathbf{currentPaint} +1 \wedge c.\mathbf{posPainting} = pa.\mathbf{currentPaint}.$
Transported for assembling when grouped.	$\forall\, c:\text{Car}, \exists tra:\text{Transporter}, aa:\text{AssemblingArea}; \neg\, \mathbf{assembled}(c) \wedge \neg\, \mathbf{isOnAA}(c,aa) \wedge \mathbf{availableTransporter}(tra) \wedge \mathbf{groupedAssembled}(c) \Rightarrow \Diamond\, \mathbf{isOnAA}(c,aa) \wedge aa.\mathbf{currentAssembled} = aa.\mathbf{currentAssembled} +1 \wedge c.\mathbf{posAssembling} = aa.\mathbf{currentAssembled}.$
The spray gun washed with solvent.	$\forall\, sg:\text{SprayGun}, \exists\, \text{painter}:Painter; \neg\, \mathbf{clean}(sg) \wedge \mathbf{has}(painter, sg) \wedge painter.\mathbf{qcarsPainted} > 0 \Rightarrow \Diamond\, \mathbf{clean}(sg) \wedge painter.\mathbf{qcarsPainted} = 0.$
The spray gun has painted when it was washed.	$\forall\, c:Car, \text{sg}:SprayGun, \exists\, \text{painter}:Painter; \mathbf{has}(painter, sg) \wedge \mathbf{clean}(sg) \Rightarrow \Diamond\, \neg\, \mathbf{clean}(sg).$


Operator
Cars grouped according spraygun limit
Cars grouped by special features
Transporter
Transported for assembling when grouped
Transported to painting when were grouped
Painter
Assembler
Cars assembled when they are grouped
The spray gun washed with solvent
The spray gun has painted when it was washed
Cars assembled when they are painted in painting area
Cars painted when they are on painting area
Cars painted when a order was received
A request is ready for delivery when a order was received

The Gas Station Problem

del Foyo, P.M.G., Salmon, A.O., Silva, J.R.; Requirements Analysis of Automated Problems Using UML/Petri Nets, Proc. of the 21st. Congress of Mech. Eng., Natal, 2011.

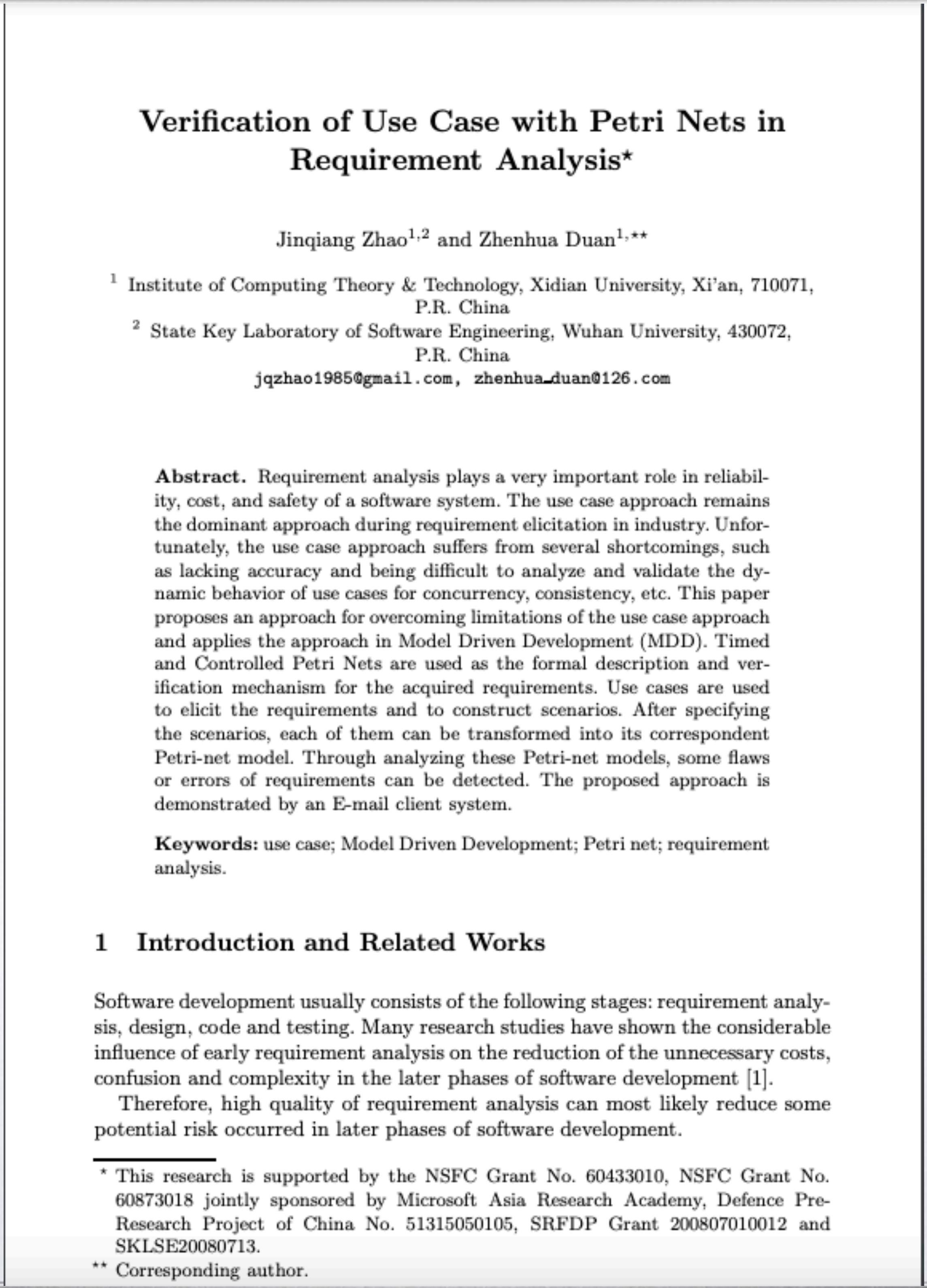




GasStation1

Name	Value
Net Name	GasStation1
Boxs	14
PseudoBoxs	0
Activities	8
Arcs	28

Baresi, L. and Pezze, M., 2001. "Improving UML with petri nets". In *UNIGRA 2001, Uniform Approaches to Graphical Process Specification Techniques (a Satellite Event of ETAPS 2001)*. Elsevier, Vol. 44 of *Electronic Notes in Theoretical Computer Science*, pp. 107–119.



Verification of Use Case with Petri Nets in Requirement Analysis*

Jinqiang Zhao[1,2] and Zhenhua Duan[1,**]

[1] Institute of Computing Theory & Technology, Xidian University, Xi'an, 710071, P.R. China
[2] State Key Laboratory of Software Engineering, Wuhan University, 430072, P.R. China
jqzhao1985@gmail.com, zhenhua_duan@126.com

Abstract. Requirement analysis plays a very important role in reliability, cost, and safety of a software system. The use case approach remains the dominant approach during requirement elicitation in industry. Unfortunately, the use case approach suffers from several shortcomings, such as lacking accuracy and being difficult to analyze and validate the dynamic behavior of use cases for concurrency, consistency, etc. This paper proposes an approach for overcoming limitations of the use case approach and applies the approach in Model Driven Development (MDD). Timed and Controlled Petri Nets are used as the formal description and verification mechanism for the acquired requirements. Use cases are used to elicit the requirements and to construct scenarios. After specifying the scenarios, each of them can be transformed into its correspondent Petri-net model. Through analyzing these Petri-net models, some flaws or errors of requirements can be detected. The proposed approach is demonstrated by an E-mail client system.

Keywords: use case; Model Driven Development; Petri net; requirement analysis.

1 Introduction and Related Works

Software development usually consists of the following stages: requirement analysis, design, code and testing. Many research studies have shown the considerable influence of early requirement analysis on the reduction of the unnecessary costs, confusion and complexity in the later phases of software development [1].

Therefore, high quality of requirement analysis can most likely reduce some potential risk occurred in later phases of software development.

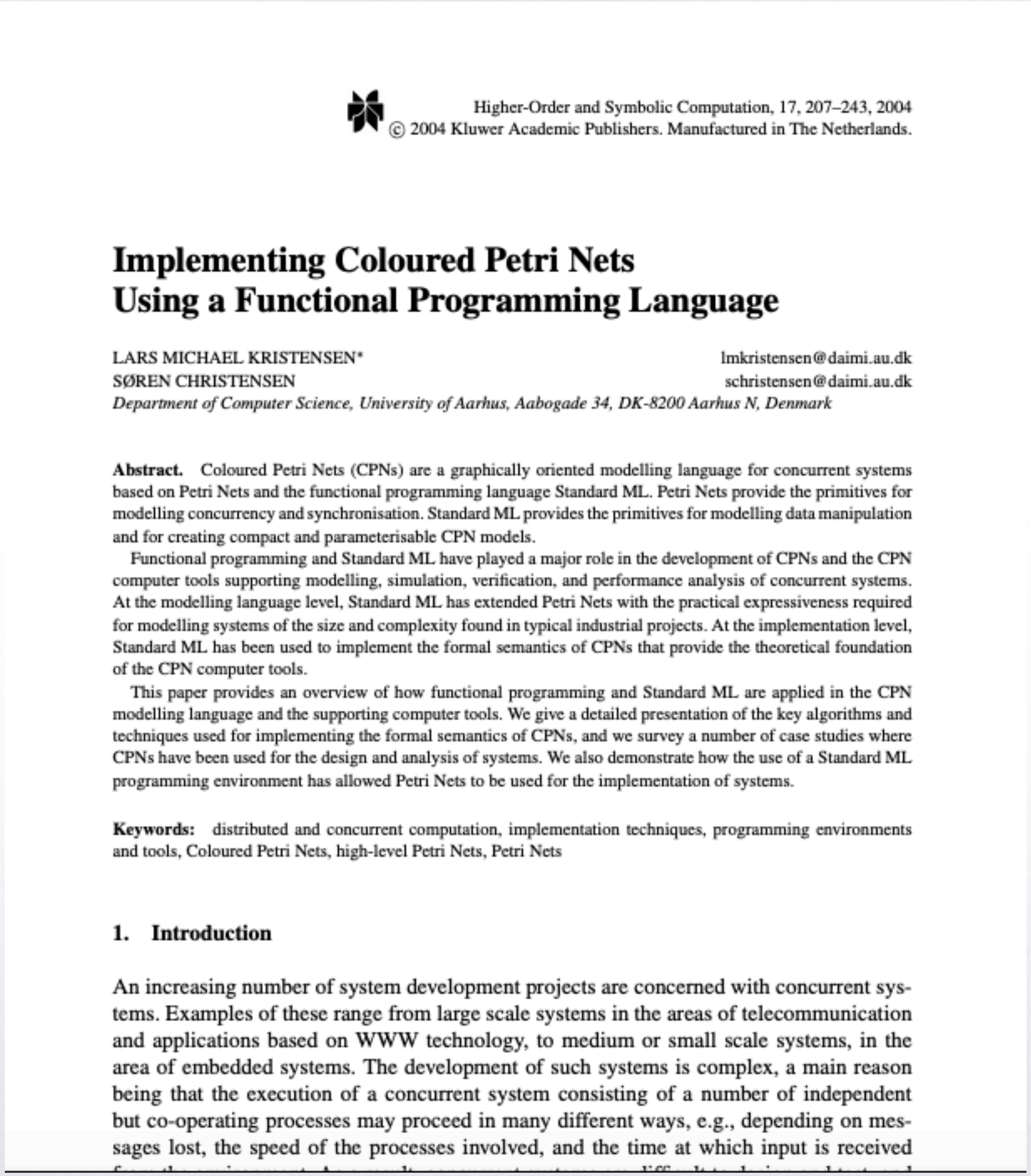
* This research is supported by the NSFC Grant No. 60433010, NSFC Grant No. 60873018 jointly sponsored by Microsoft Asia Research Academy, Defence Pre-Research Project of China No. 51315050105, SRFDP Grant 200807010012 and SKLSE20080713.
** Corresponding author.

Theoretical and practical reasons to use CPN Nets

Improve abstraction (functional programming)

Improve the formal semantics of PN

Sound execution based sound support tools



Higher-Order and Symbolic Computation, 17, 207–243, 2004
© 2004 Kluwer Academic Publishers. Manufactured in The Netherlands.

Implementing Coloured Petri Nets Using a Functional Programming Language

LARS MICHAEL KRISTENSEN* lmkristensen@daimi.au.dk
SØREN CHRISTENSEN schristensen@daimi.au.dk
Department of Computer Science, University of Aarhus, Aabogade 34, DK-8200 Aarhus N, Denmark

Abstract. Coloured Petri Nets (CPNs) are a graphically oriented modelling language for concurrent systems based on Petri Nets and the functional programming language Standard ML. Petri Nets provide the primitives for modelling concurrency and synchronisation. Standard ML provides the primitives for modelling data manipulation and for creating compact and parameterisable CPN models.

Functional programming and Standard ML have played a major role in the development of CPNs and the CPN computer tools supporting modelling, simulation, verification, and performance analysis of concurrent systems. At the modelling language level, Standard ML has extended Petri Nets with the practical expressiveness required for modelling systems of the size and complexity found in typical industrial projects. At the implementation level, Standard ML has been used to implement the formal semantics of CPNs that provide the theoretical foundation of the CPN computer tools.

This paper provides an overview of how functional programming and Standard ML are applied in the CPN modelling language and the supporting computer tools. We give a detailed presentation of the key algorithms and techniques used for implementing the formal semantics of CPNs, and we survey a number of case studies where CPNs have been used for the design and analysis of systems. We also demonstrate how the use of a Standard ML programming environment has allowed Petri Nets to be used for the implementation of systems.

Keywords: distributed and concurrent computation, implementation techniques, programming environments and tools, Coloured Petri Nets, high-level Petri Nets, Petri Nets

1. Introduction

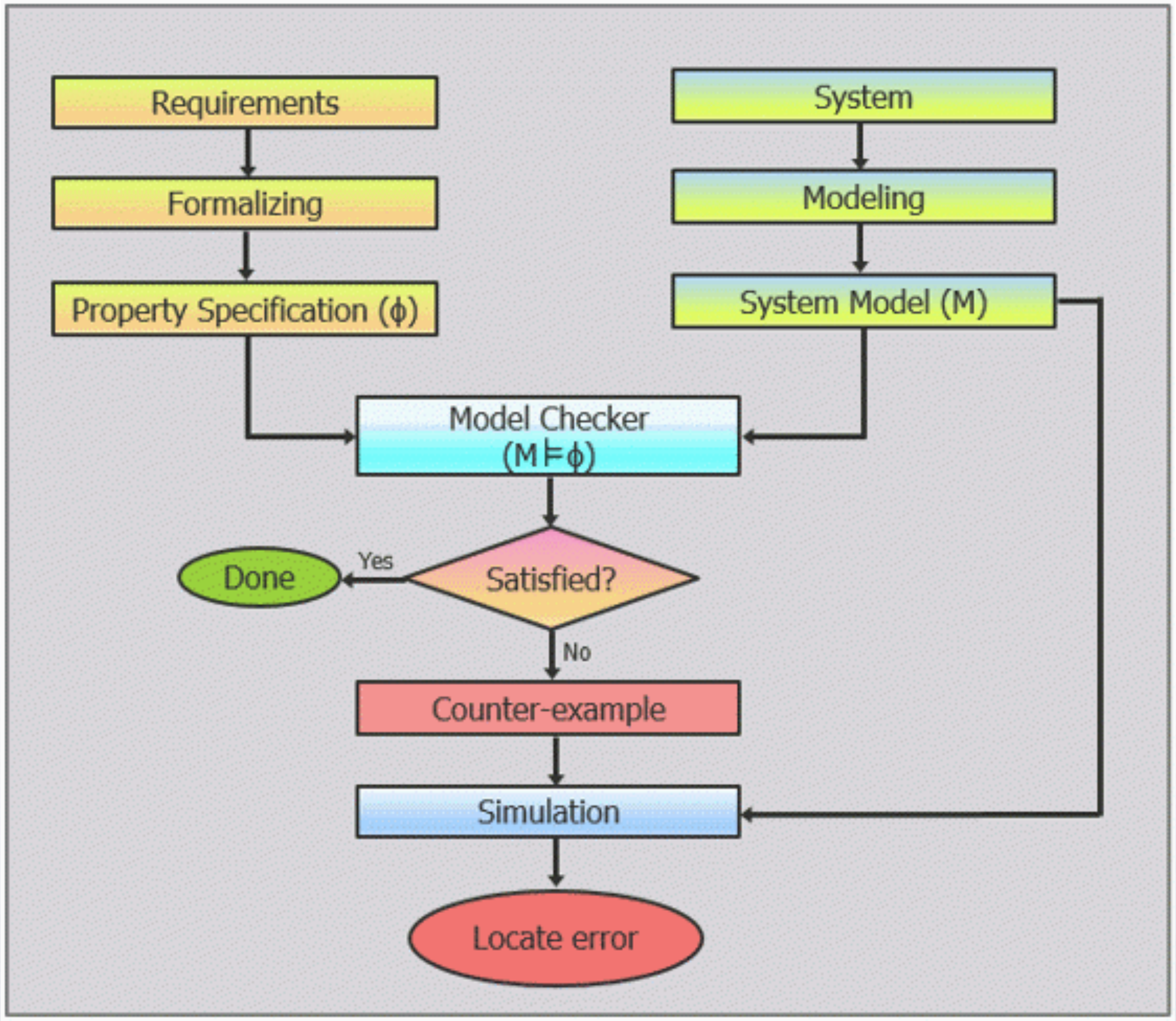
An increasing number of system development projects are concerned with concurrent systems. Examples of these range from large scale systems in the areas of telecommunication and applications based on WWW technology, to medium or small scale systems, in the area of embedded systems. The development of such systems is complex, a main reason being that the execution of a concurrent system consisting of a number of independent but co-operating processes may proceed in many different ways, e.g., depending on messages lost, the speed of the processes involved, and the time at which input is received

What are the drawbacks?

CPN was designed to improve semantic analysis but not to fit together with classic PN analysis.

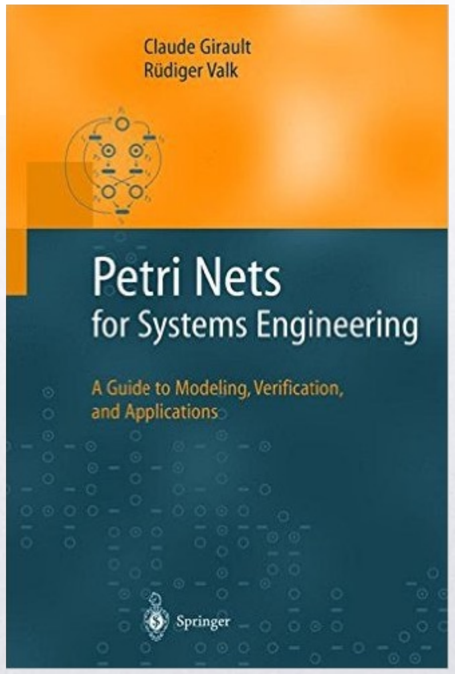
Automation of the analysis of some structural and behavioral properties was lost.

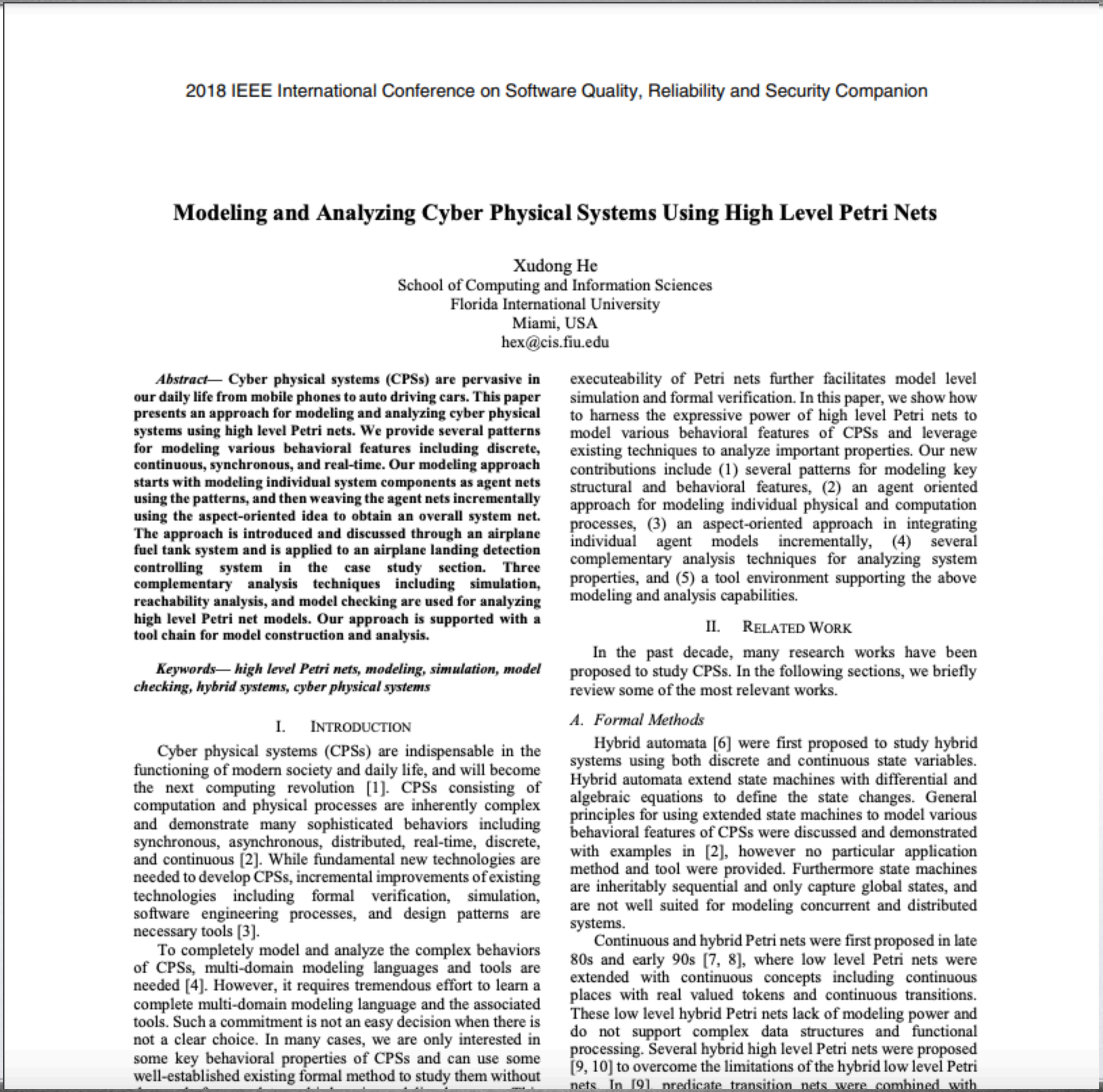
Improve the design approach but detach from control and direct implementation.


Requirements
Formalizing
Property Specification (ϕ)
System
Modeling
System Model (M)
Model Checker (M⊨ϕ)
Done
Yes
Satisfied?
No
Counter-example
Simulation
Locate error

Matching by construction

1. Create the set of places used for the net model.
2. Design constraints describing the behaviour and the structure of the solution, which ensure at least the safety properties of the specification.
3. Add all transitions that do not violate the constraints.
4. Prove the dynamic properties.





2018 IEEE International Conference on Software Quality, Reliability and Security Companion

Modeling and Analyzing Cyber Physical Systems Using High Level Petri Nets

Xudong He
School of Computing and Information Sciences
Florida International University
Miami, USA
hex@cis.fiu.edu

***Abstract*— Cyber physical systems (CPSs) are pervasive in our daily life from mobile phones to auto driving cars. This paper presents an approach for modeling and analyzing cyber physical systems using high level Petri nets. We provide several patterns for modeling various behavioral features including discrete, continuous, synchronous, and real-time. Our modeling approach starts with modeling individual system components as agent nets using the patterns, and then weaving the agent nets incrementally using the aspect-oriented idea to obtain an overall system net. The approach is introduced and discussed through an airplane fuel tank system and is applied to an airplane landing detection controlling system in the case study section. Three complementary analysis techniques including simulation, reachability analysis, and model checking are used for analyzing high level Petri net models. Our approach is supported with a tool chain for model construction and analysis.**

Keywords*— high level Petri nets, modeling, simulation, model checking, hybrid systems, cyber physical systems

I. Introduction

Cyber physical systems (CPSs) are indispensable in the functioning of modern society and daily life, and will become the next computing revolution [1]. CPSs consisting of computation and physical processes are inherently complex and demonstrate many sophisticated behaviors including synchronous, asynchronous, distributed, real-time, discrete, and continuous [2]. While fundamental new technologies are needed to develop CPSs, incremental improvements of existing technologies including formal verification, simulation, software engineering processes, and design patterns are necessary tools [3].

To completely model and analyze the complex behaviors of CPSs, multi-domain modeling languages and tools are needed [4]. However, it requires tremendous effort to learn a complete multi-domain modeling language and the associated tools. Such a commitment is not an easy decision when there is not a clear choice. In many cases, we are only interested in some key behavioral properties of CPSs and can use some well-established existing formal method to study them without

executeability of Petri nets further facilitates model level simulation and formal verification. In this paper, we show how to harness the expressive power of high level Petri nets to model various behavioral features of CPSs and leverage existing techniques to analyze important properties. Our new contributions include (1) several patterns for modeling key structural and behavioral features, (2) an agent oriented approach for modeling individual physical and computation processes, (3) an aspect-oriented approach in integrating individual agent models incrementally, (4) several complementary analysis techniques for analyzing system properties, and (5) a tool environment supporting the above modeling and analysis capabilities.

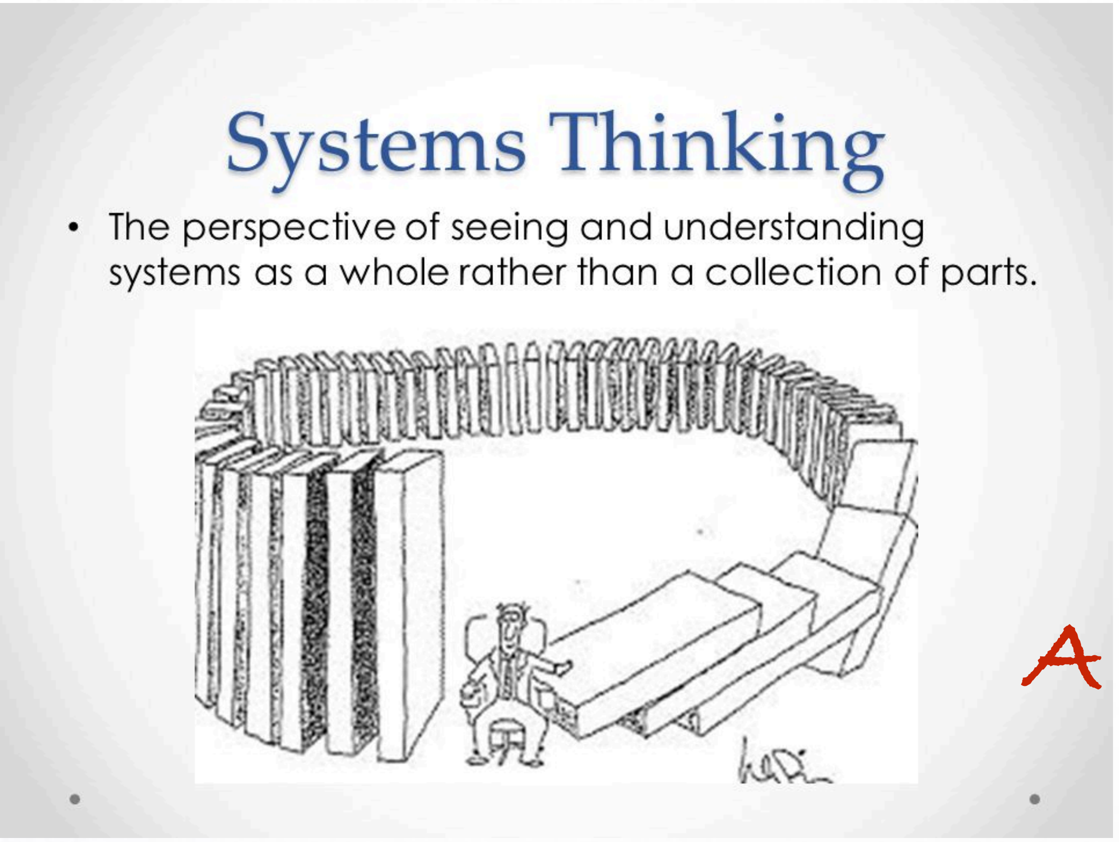
II. Related Work

In the past decade, many research works have been proposed to study CPSs. In the following sections, we briefly review some of the most relevant works.

A. Formal Methods

Hybrid automata [6] were first proposed to study hybrid systems using both discrete and continuous state variables. Hybrid automata extend state machines with differential and algebraic equations to define the state changes. General principles for using extended state machines to model various behavioral features of CPSs were discussed and demonstrated with examples in [2], however no particular application method and tool were provided. Furthermore state machines are inheritably sequential and only capture global states, and are not well suited for modeling concurrent and distributed systems.

Continuous and hybrid Petri nets were first proposed in late 80s and early 90s [7, 8], where low level Petri nets were extended with continuous concepts including continuous places with real valued tokens and continuous transitions. These low level hybrid Petri nets lack of modeling power and do not support complex data structures and functional processing. Several hybrid high level Petri nets were proposed [9, 10] to overcome the limitations of the hybrid low level Petri nets. In [9], predicate transition nets were combined with



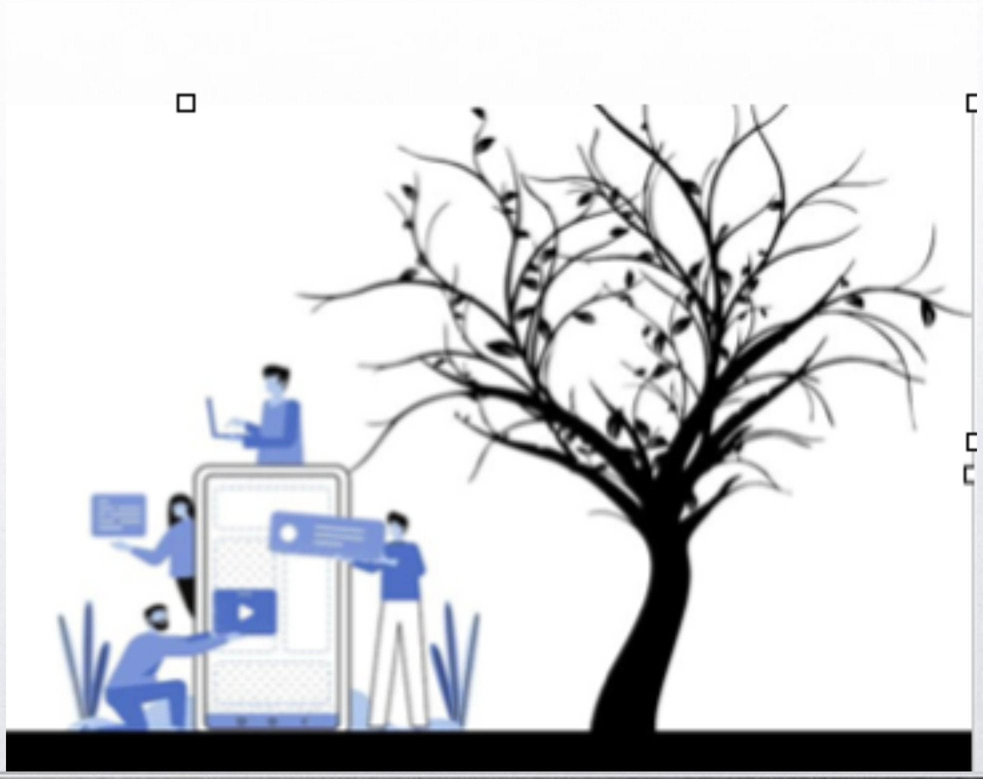
Systems Thinking

- The perspective of seeing and understanding systems as a whole rather than a collection of parts.

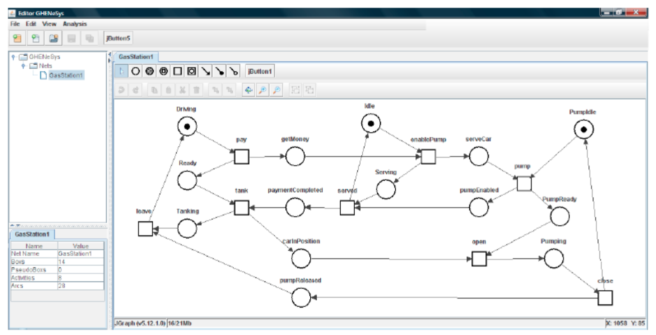
Look at the forest first and refine,

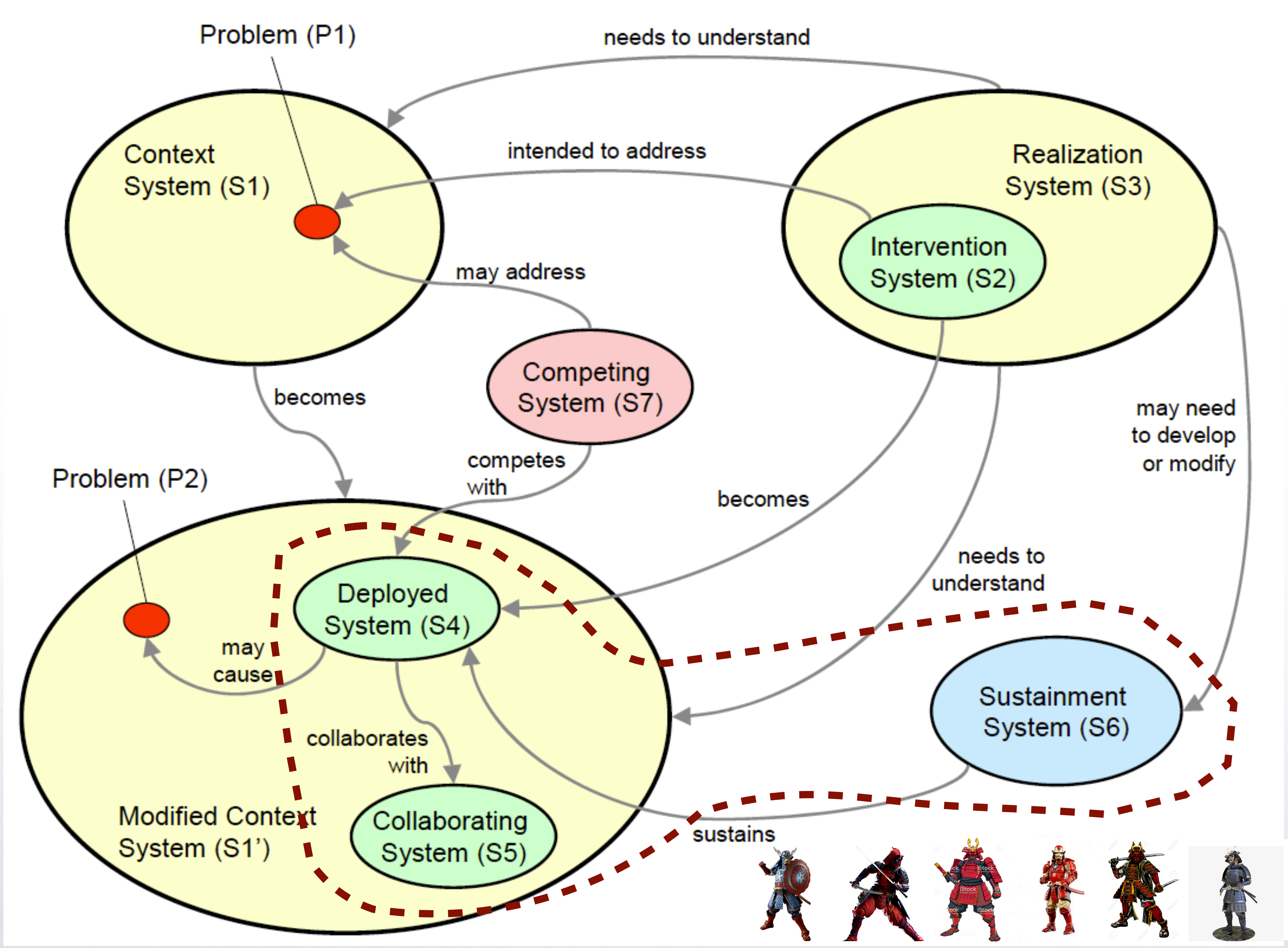
A false dilema!?

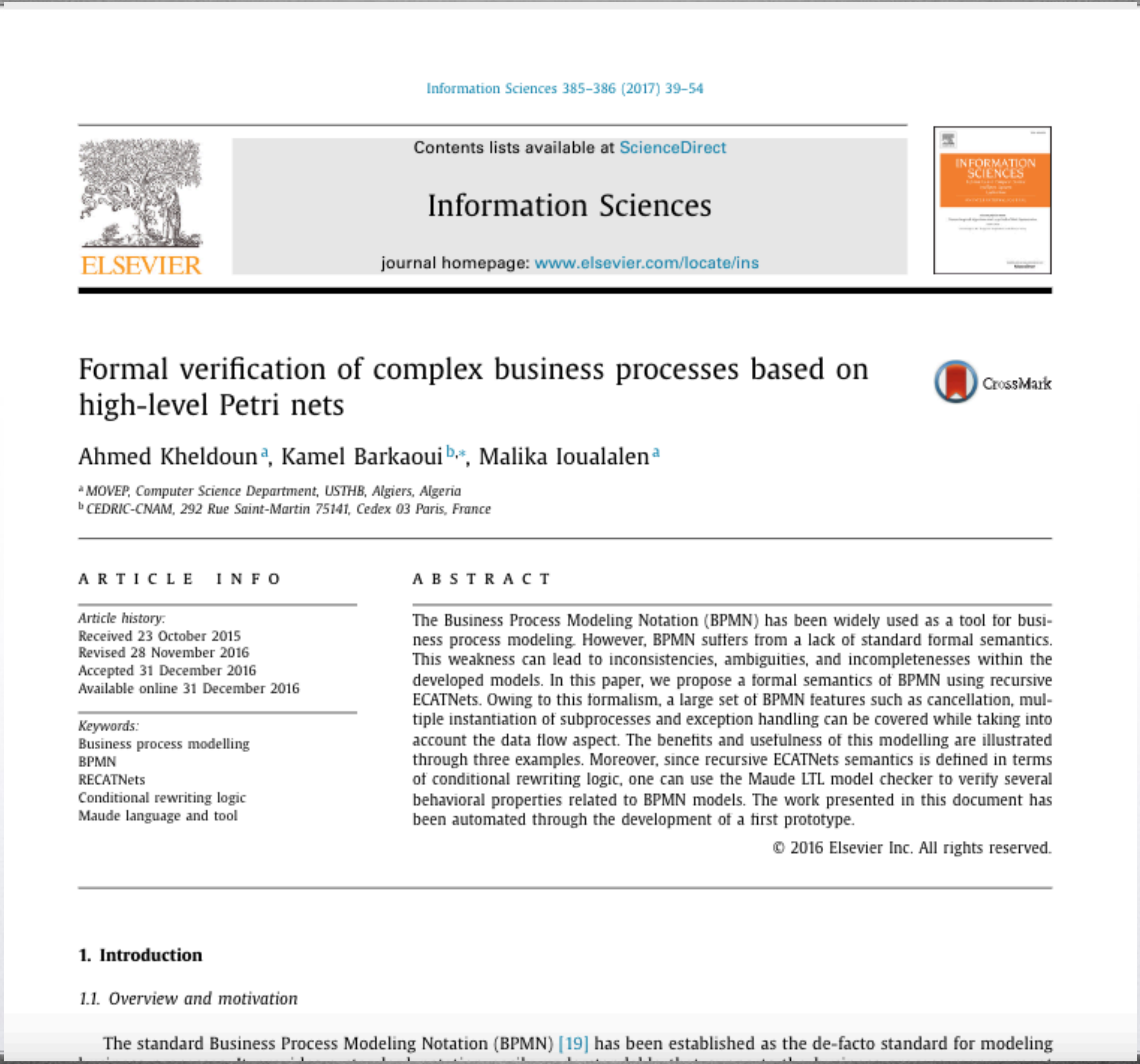
Look at the trees and compose.



The Seven Samurais







Information Sciences 385–386 (2017) 39–54

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Formal verification of complex business processes based on high-level Petri nets

Ahmed Kheldoun[a], Kamel Barkaoui[b,*], Malika Ioualalen[a]

[a] MOVEP, Computer Science Department, USTHB, Algiers, Algeria
[b] CEDRIC-CNAM, 292 Rue Saint-Martin 75141, Cedex 03 Paris, France

ARTICLE INFO

Article history:
Received 23 October 2015
Revised 28 November 2016
Accepted 31 December 2016
Available online 31 December 2016

Keywords:
Business process modelling
BPMN
RECATNets
Conditional rewriting logic
Maude language and tool

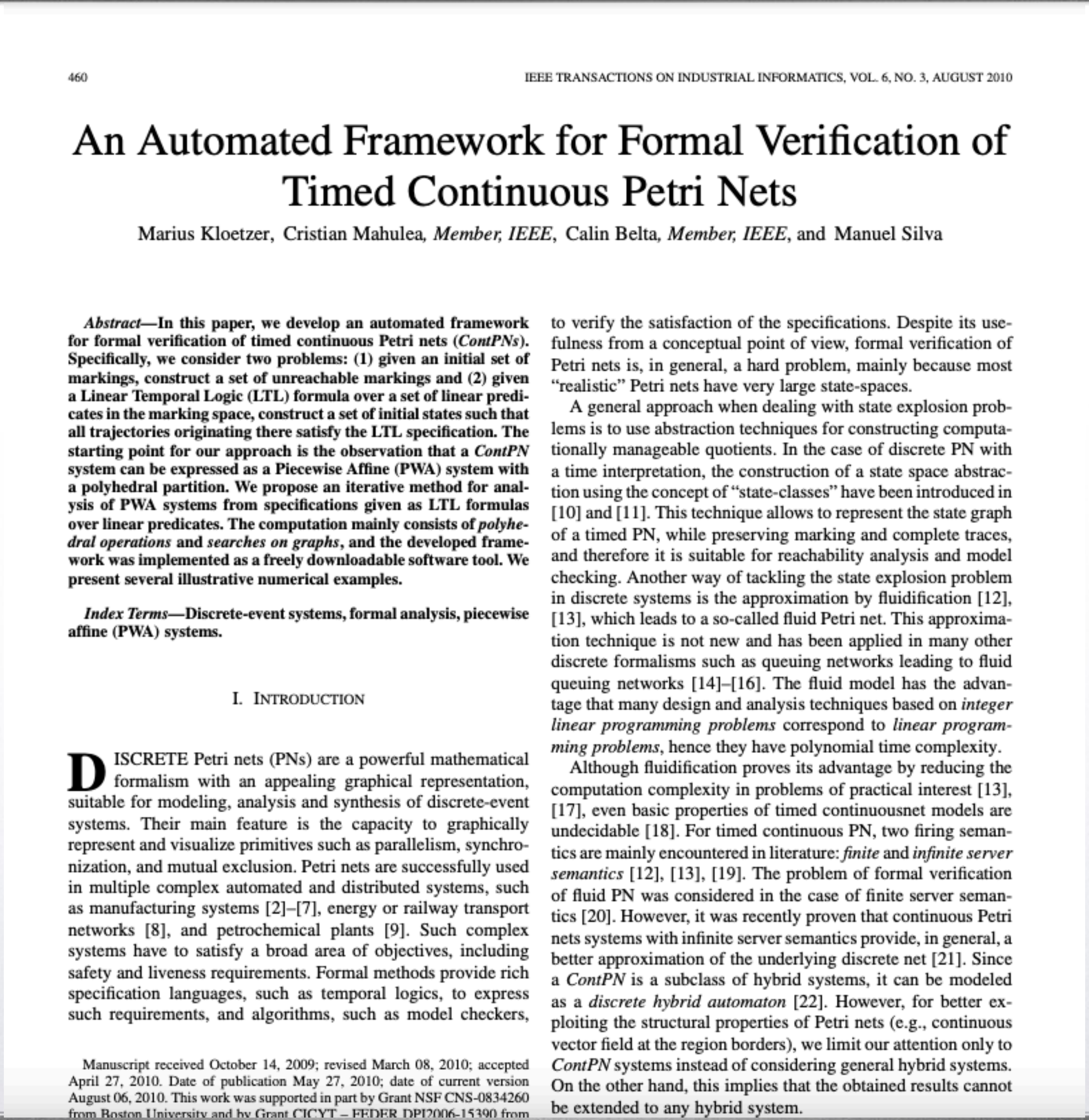
ABSTRACT

The Business Process Modeling Notation (BPMN) has been widely used as a tool for business process modeling. However, BPMN suffers from a lack of standard formal semantics. This weakness can lead to inconsistencies, ambiguities, and incompletenesses within the developed models. In this paper, we propose a formal semantics of BPMN using recursive ECATNets. Owing to this formalism, a large set of BPMN features such as cancellation, multiple instantiation of subprocesses and exception handling can be covered while taking into account the data flow aspect. The benefits and usefulness of this modelling are illustrated through three examples. Moreover, since recursive ECATNets semantics is defined in terms of conditional rewriting logic, one can use the Maude LTL model checker to verify several behavioral properties related to BPMN models. The work presented in this document has been automated through the development of a first prototype.

1. Introduction

1.1. Overview and motivation

The standard Business Process Modeling Notation (BPMN) [19] has been established as the de-facto standard for modeling



An Automated Framework for Formal Verification of Timed Continuous Petri Nets

Marius Kloetzer, Cristian Mahulea, *Member, IEEE*, Calin Belta, *Member, IEEE*, and Manuel Silva

***Abstract*—In this paper, we develop an automated framework for formal verification of timed continuous Petri nets (*ContPNs*). Specifically, we consider two problems: (1) given an initial set of markings, construct a set of unreachable markings and (2) given a Linear Temporal Logic (LTL) formula over a set of linear predicates in the marking space, construct a set of initial states such that all trajectories originating there satisfy the LTL specification. The starting point for our approach is the observation that a *ContPN* system can be expressed as a Piecewise Affine (PWA) system with a polyhedral partition. We propose an iterative method for analysis of PWA systems from specifications given as LTL formulas over linear predicates. The computation mainly consists of *polyhedral operations* and *searches on graphs*, and the developed framework was implemented as a freely downloadable software tool. We present several illustrative numerical examples.**

***Index Terms*—Discrete-event systems, formal analysis, piecewise affine (PWA) systems.**

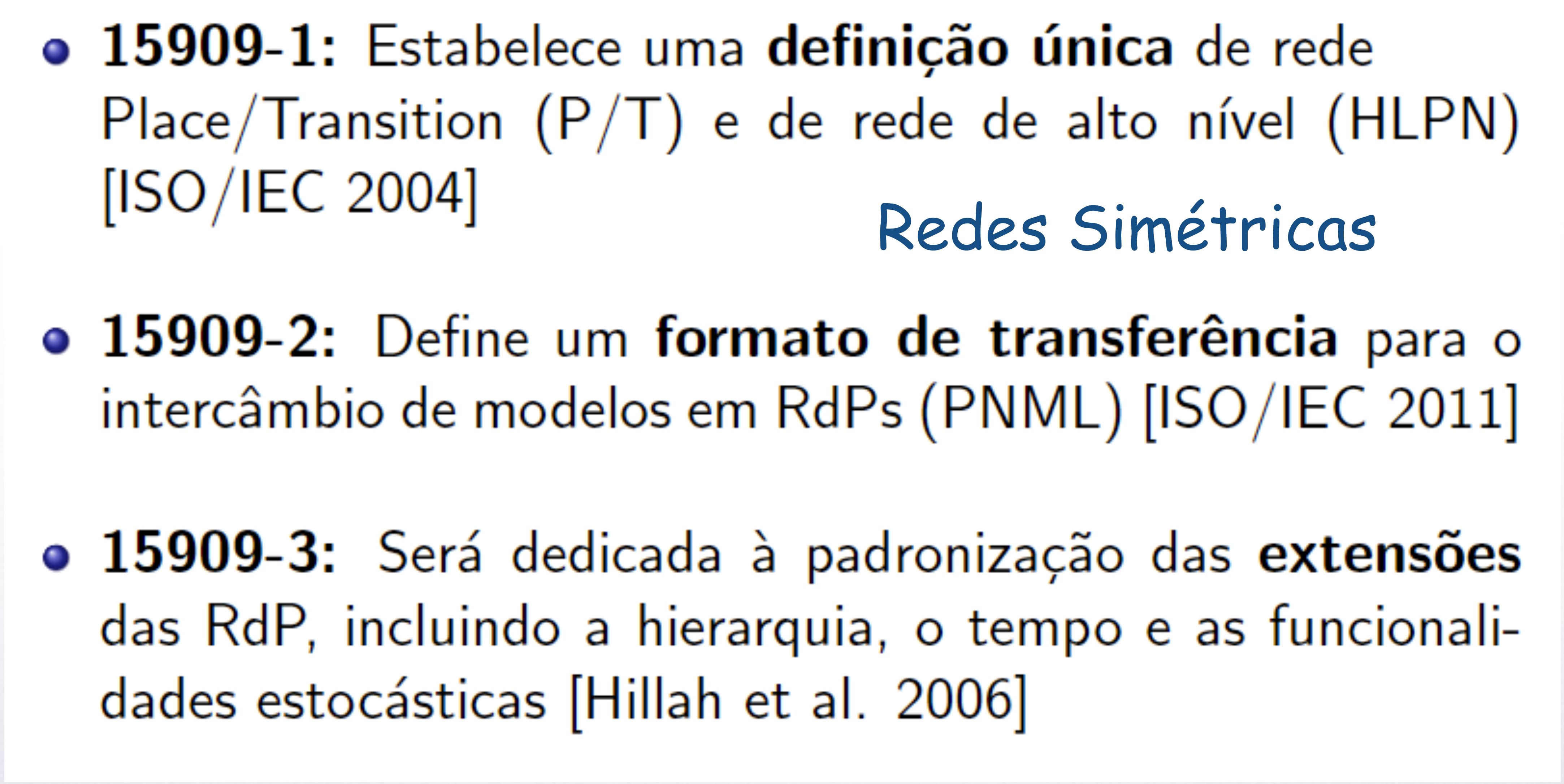
I. Introduction

DISCRETE Petri nets (PNs) are a powerful mathematical formalism with an appealing graphical representation, suitable for modeling, analysis and synthesis of discrete-event systems. Their main feature is the capacity to graphically represent and visualize primitives such as parallelism, synchronization, and mutual exclusion. Petri nets are successfully used in multiple complex automated and distributed systems, such as manufacturing systems [2]–[7], energy or railway transport networks [8], and petrochemical plants [9]. Such complex systems have to satisfy a broad area of objectives, including safety and liveness requirements. Formal methods provide rich specification languages, such as temporal logics, to express such requirements, and algorithms, such as model checkers, to verify the satisfaction of the specifications. Despite its usefulness from a conceptual point of view, formal verification of Petri nets is, in general, a hard problem, mainly because most "realistic" Petri nets have very large state-spaces.

A general approach when dealing with state explosion problems is to use abstraction techniques for constructing computationally manageable quotients. In the case of discrete PN with a time interpretation, the construction of a state space abstraction using the concept of "state-classes" have been introduced in [10] and [11]. This technique allows to represent the state graph of a timed PN, while preserving marking and complete traces, and therefore it is suitable for reachability analysis and model checking. Another way of tackling the state explosion problem in discrete systems is the approximation by fluidification [12], [13], which leads to a so-called fluid Petri net. This approximation technique is not new and has been applied in many other discrete formalisms such as queuing networks leading to fluid queuing networks [14]–[16]. The fluid model has the advantage that many design and analysis techniques based on *integer linear programming problems* correspond to *linear programming problems*, hence they have polynomial time complexity.

Although fluidification proves its advantage by reducing the computation complexity in problems of practical interest [13], [17], even basic properties of timed continuousnet models are undecidable [18]. For timed continuous PN, two firing semantics are mainly encountered in literature: *finite* and *infinite server semantics* [12], [13], [19]. The problem of formal verification of fluid PN was considered in the case of finite server semantics [20]. However, it was recently proven that continuous Petri nets systems with infinite server semantics provide, in general, a better approximation of the underlying discrete net [21]. Since a *ContPN* is a subclass of hybrid systems, it can be modeled as a *discrete hybrid automaton* [22]. However, for better exploiting the structural properties of Petri nets (e.g., continuous vector field at the region borders), we limit our attention only to *ContPN* systems instead of considering general hybrid systems. On the other hand, this implies that the obtained results cannot be extended to any hybrid system.

Manuscript received October 14, 2009; revised March 08, 2010; accepted April 27, 2010. Date of publication May 27, 2010; date of current version August 06, 2010. This work was supported in part by Grant NSF CNS-0834260 from Boston University and by Grant CICYT – FEDER DPI2006-15390 from



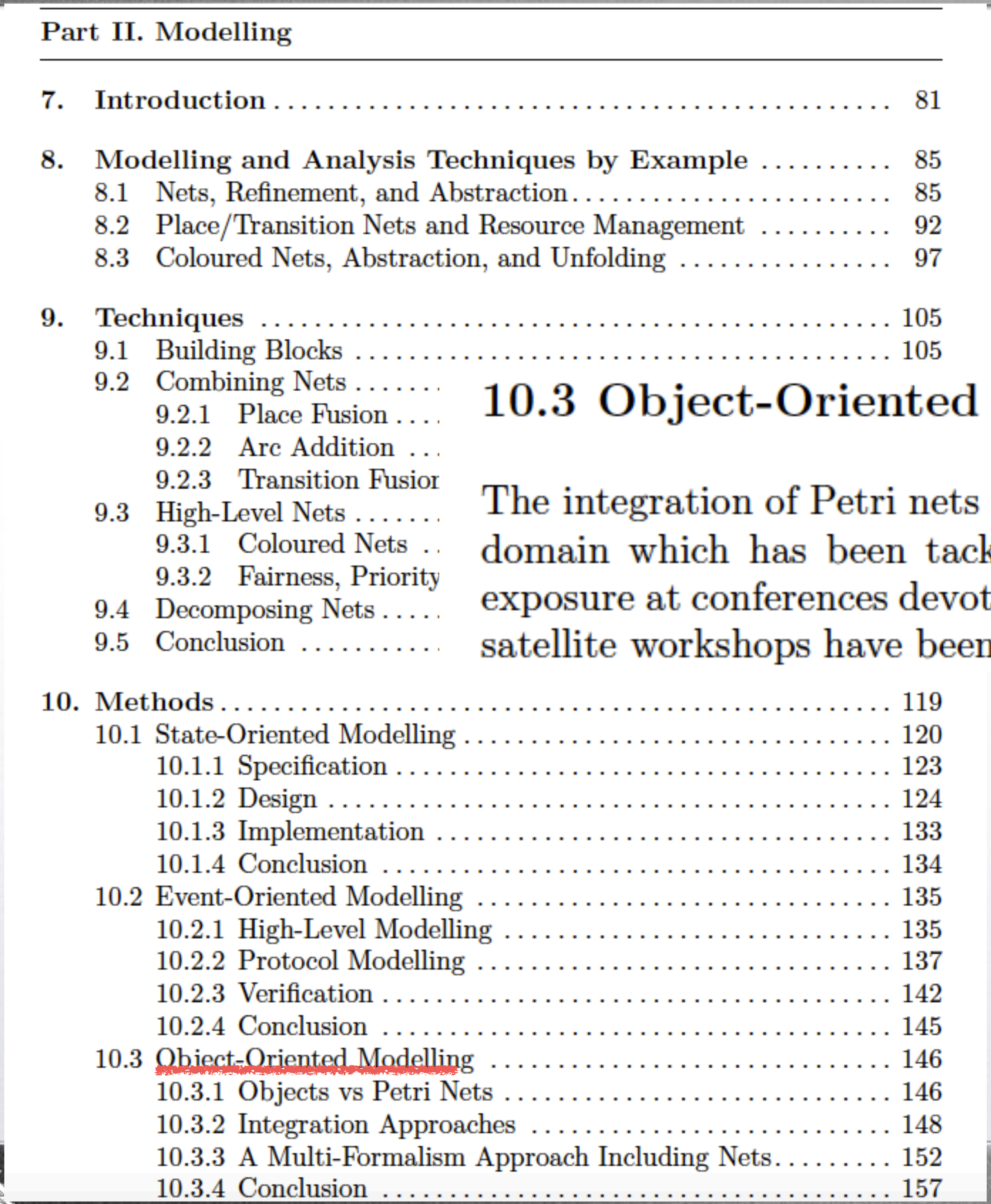
- **15909-1:** Estabelece uma **definição única** de rede Place/Transition (P/T) e de rede de alto nível (HLPN) [ISO/IEC 2004]

Redes Simétricas

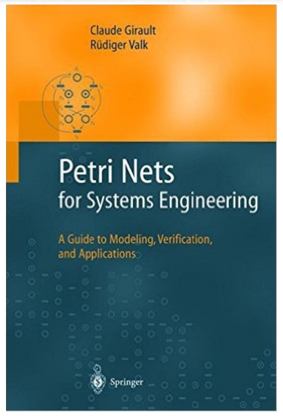
- **15909-2:** Define um **formato de transferência** para o intercâmbio de modelos em RdPs (PNML) [ISO/IEC 2011]

- **15909-3:** Será dedicada à padronização das **extensões** das RdP, incluindo a hierarquia, o tempo e as funcionalidades estocásticas [Hillah et al. 2006]

Redes Orientadas a Objetos



Part II. Modelling

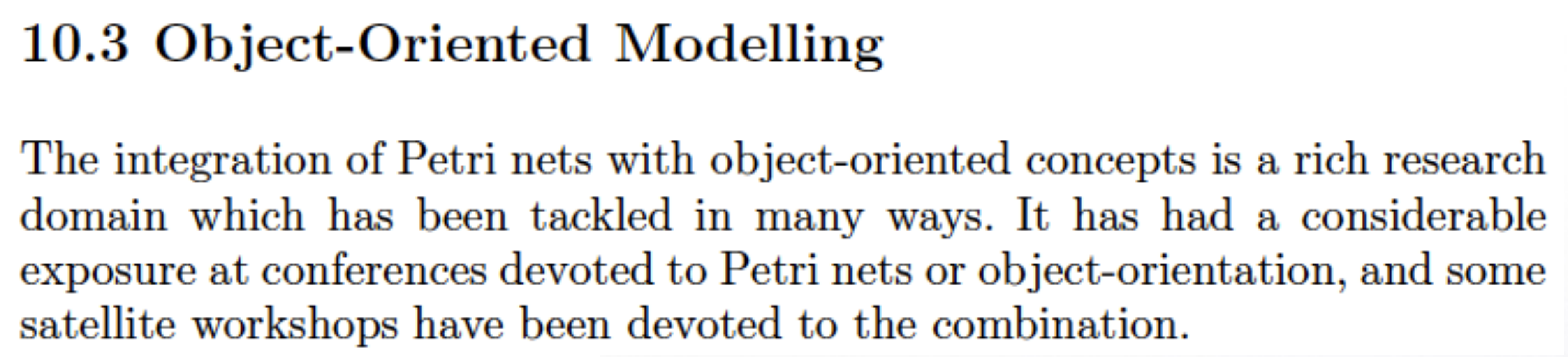


Claude Girault
Rüdiger Valk

Petri Nets
for Systems Engineering

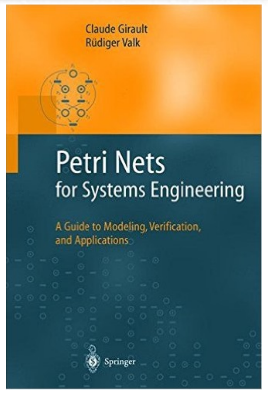
A Guide to Modeling, Verification, and Applications

Springer



10.3 Object-Oriented Modelling

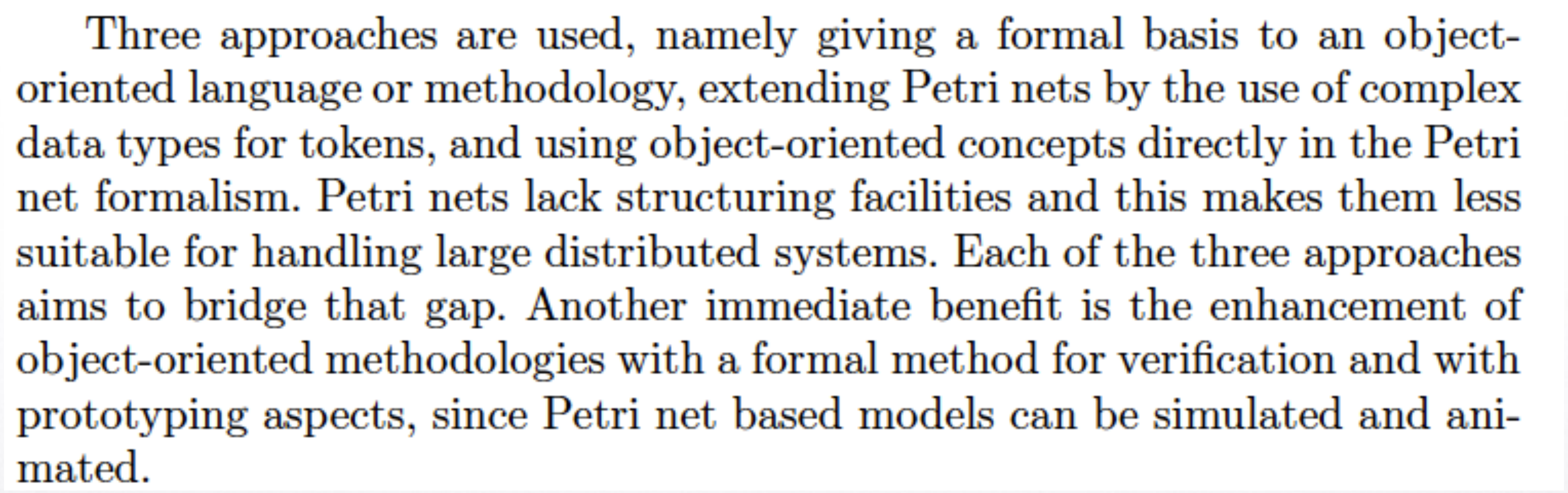
The integration of Petri nets with object-oriented concepts is a rich research domain which has been tackled in many ways. It has had a considerable exposure at conferences devoted to Petri nets or object-orientation, and some satellite workshops have been devoted to the combination.

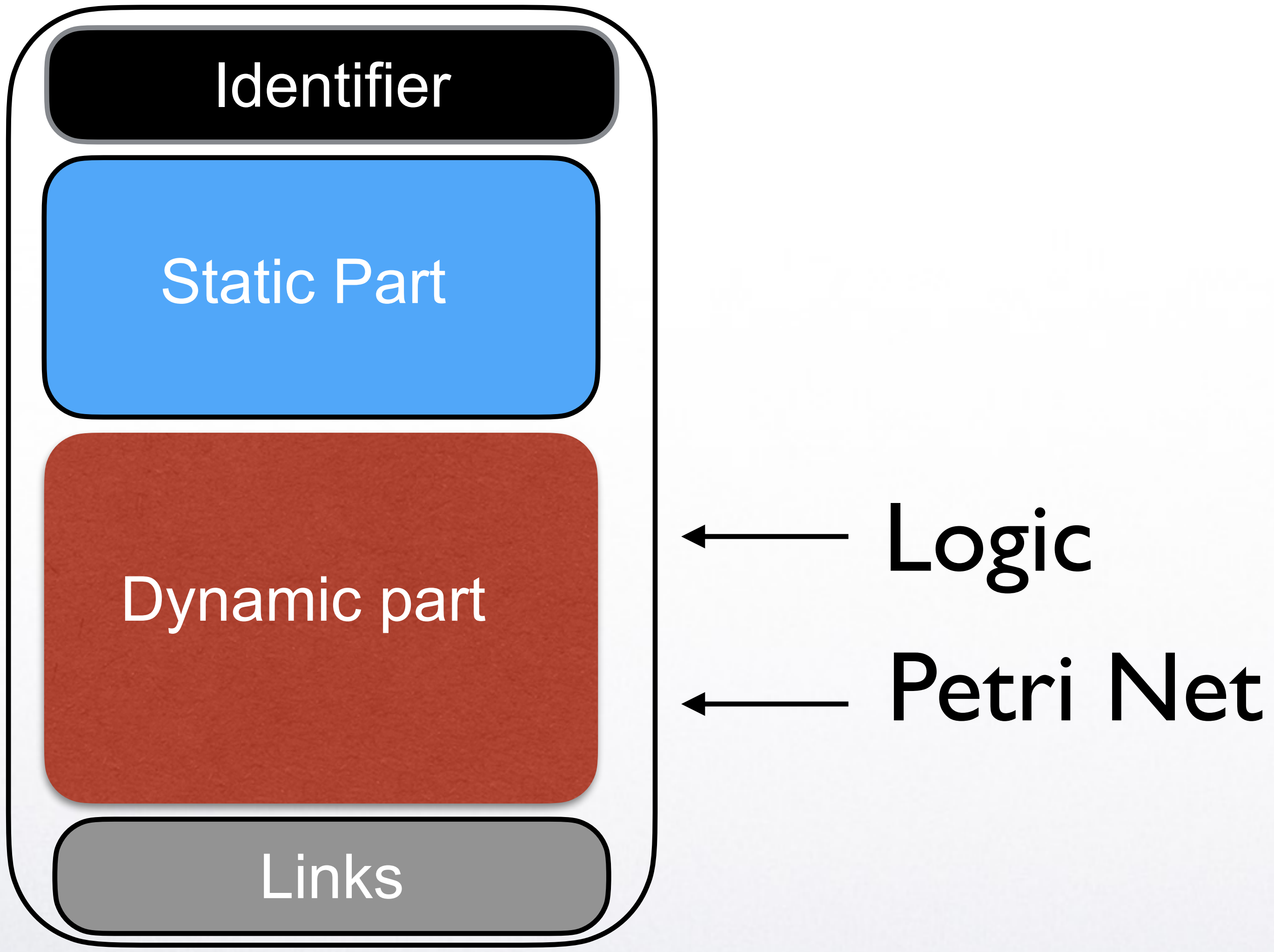

Claude Girault
Rüdiger Valk

Petri Nets
for Systems Engineering

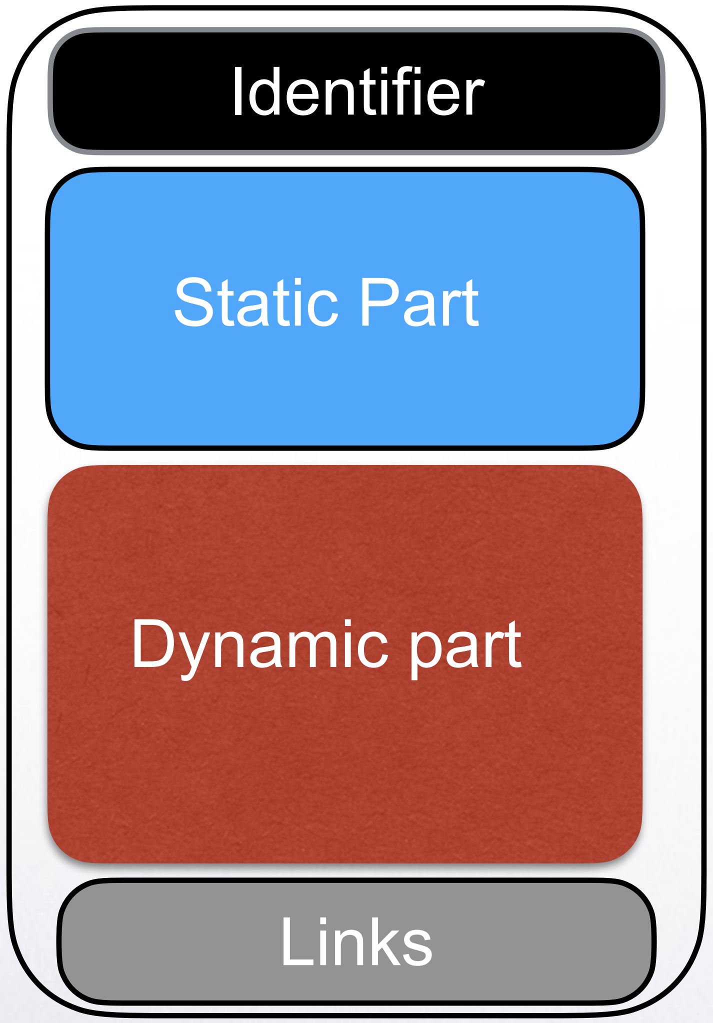
A Guide to Modeling, Verification, and Applications

Springer


Three approaches are used, namely giving a formal basis to an object-oriented language or methodology, extending Petri nets by the use of complex data types for tokens, and using object-oriented concepts directly in the Petri net formalism. Petri nets lack structuring facilities and this makes them less suitable for handling large distributed systems. Each of the three approaches aims to bridge that gap. Another immediate benefit is the enhancement of object-oriented methodologies with a formal method for verification and with prototyping aspects, since Petri net based models can be simulated and animated.

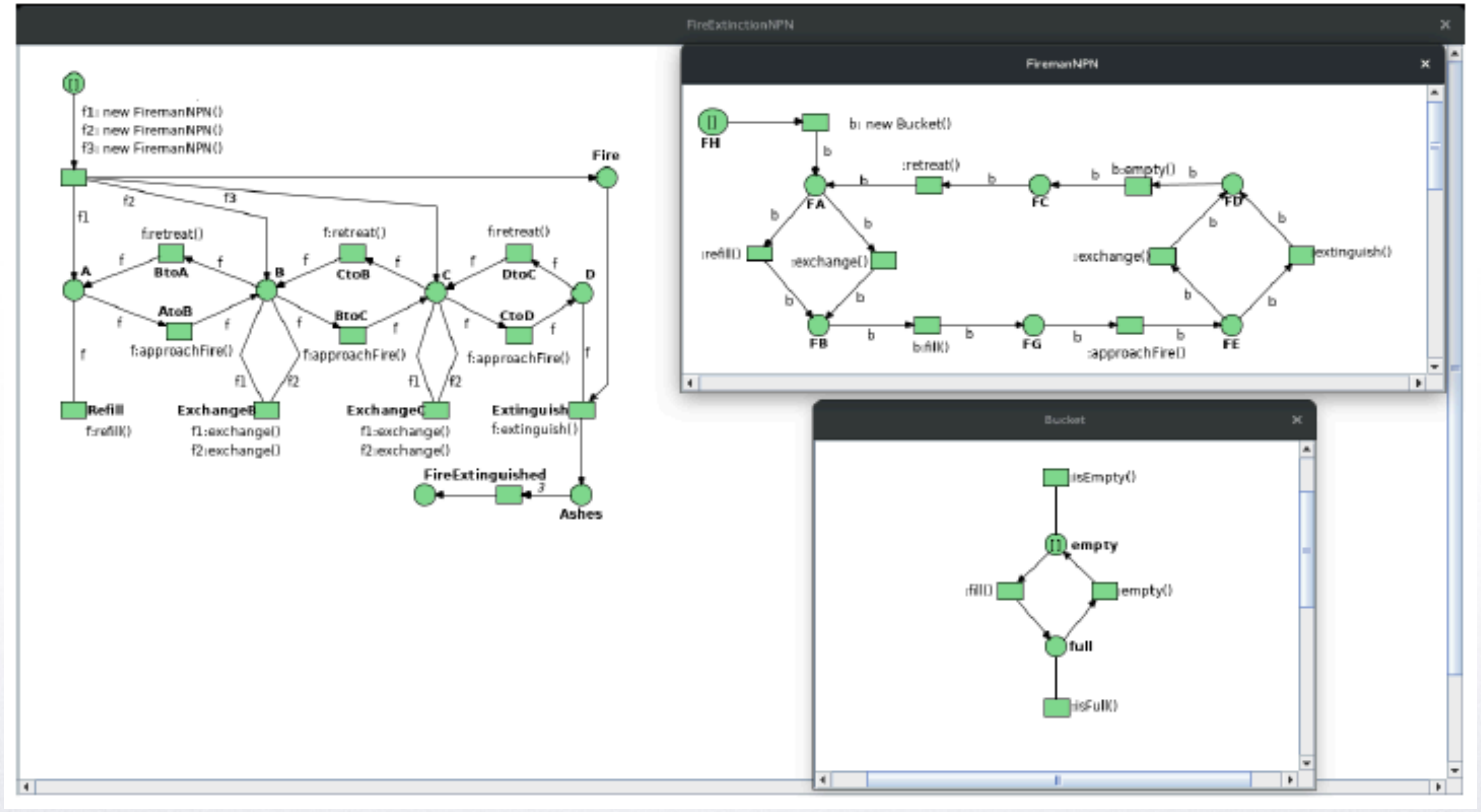

Identifier
Static Part
Dynamic part
Links
Logic
Petri Net

Tagged Petri net

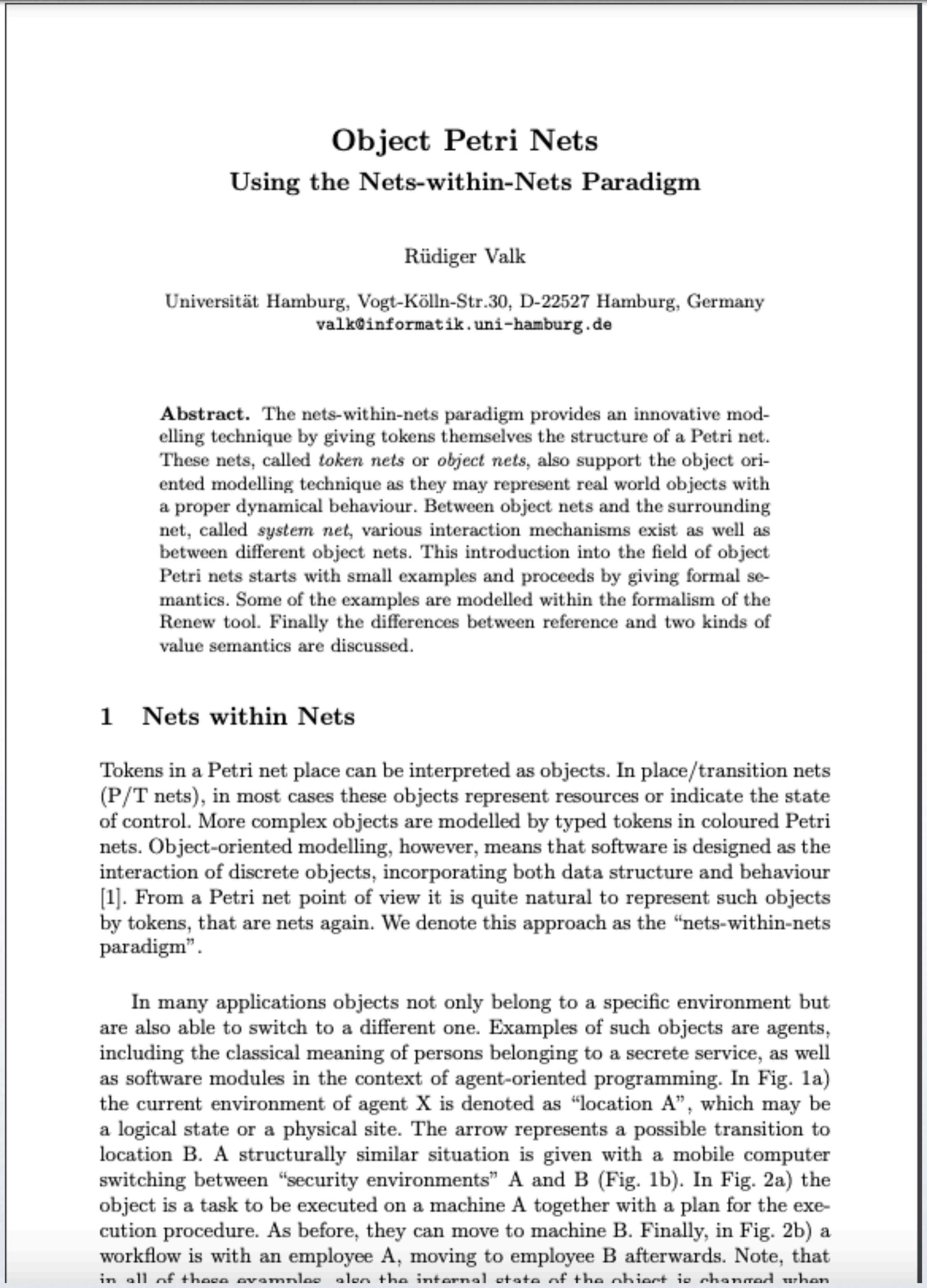


A tagged object could be used to represent places, transitions and arcs in a Petri net.

What about tokens?



FireExtinctionNPN
f1: new FiremanNPN()
f2: new FiremanNPN()
f3: new FiremanNPN()
Fire
A
B
C
D
BtoA
AtoB
CtoB
BtoC
DtoC
CtoD
f:retreat()
f:approachFire()
Refill
f:refill()
ExchangeB
f1:exchange()
f2:exchange()
ExchangeC
f1:exchange()
f2:exchange()
Extinguish
f:extinguish()
FireExtinguished
Ashes
FiremanNPN
FH
b: new Bucket()
FA
FB
FC
FD
FE
FG
:retreat()
b:empty()
:refill()
:exchange()
:extinguish()
b:fill()
:approachFire()
Bucket
:isEmpty()
empty
:fill()
:empty()
full
:isFull()



Object Petri Nets
Using the Nets-within-Nets Paradigm

Rüdiger Valk

Universität Hamburg, Vogt-Kölln-Str.30, D-22527 Hamburg, Germany
valk@informatik.uni-hamburg.de

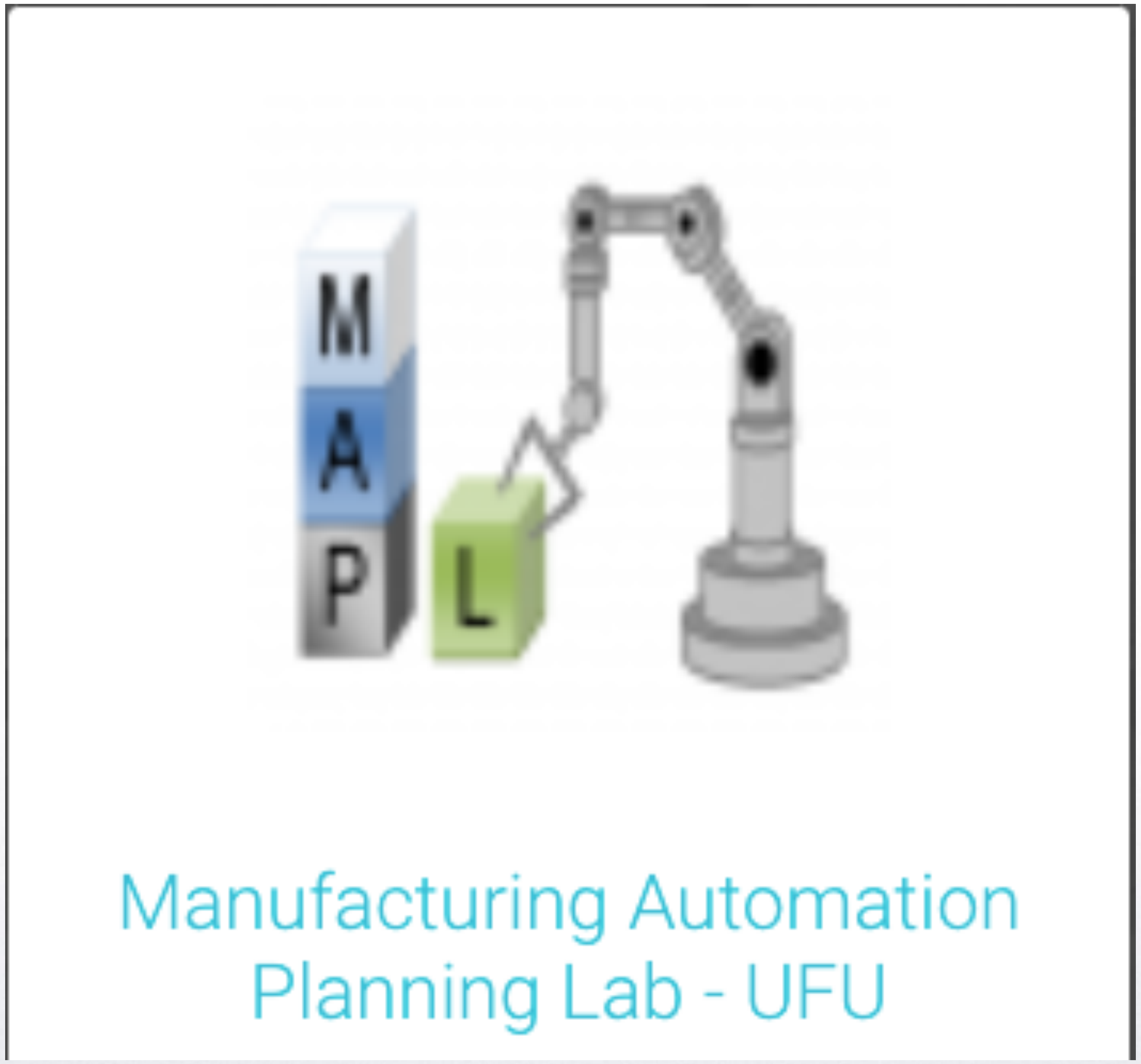
Abstract. The nets-within-nets paradigm provides an innovative modelling technique by giving tokens themselves the structure of a Petri net. These nets, called *token nets* or *object nets*, also support the object oriented modelling technique as they may represent real world objects with a proper dynamical behaviour. Between object nets and the surrounding net, called *system net*, various interaction mechanisms exist as well as between different object nets. This introduction into the field of object Petri nets starts with small examples and proceeds by giving formal semantics. Some of the examples are modelled within the formalism of the Renew tool. Finally the differences between reference and two kinds of value semantics are discussed.

1 Nets within Nets

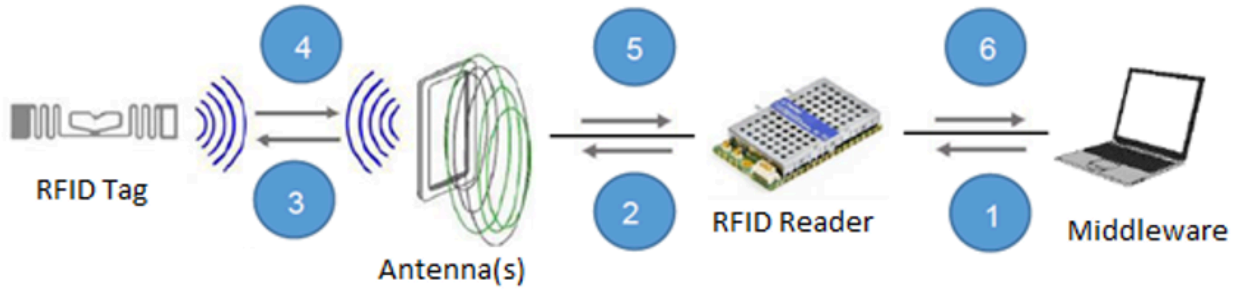
Tokens in a Petri net place can be interpreted as objects. In place/transition nets (P/T nets), in most cases these objects represent resources or indicate the state of control. More complex objects are modelled by typed tokens in coloured Petri nets. Object-oriented modelling, however, means that software is designed as the interaction of discrete objects, incorporating both data structure and behaviour [1]. From a Petri net point of view it is quite natural to represent such objects by tokens, that are nets again. We denote this approach as the "nets-within-nets paradigm".

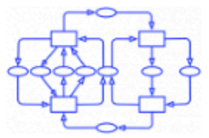
In many applications objects not only belong to a specific environment but are also able to switch to a different one. Examples of such objects are agents, including the classical meaning of persons belonging to a secrete service, as well as software modules in the context of agent-oriented programming. In Fig. 1a) the current environment of agent X is denoted as "location A", which may be a logical state or a physical site. The arrow represents a possible transition to location B. A structurally similar situation is given with a mobile computer switching between "security environments" A and B (Fig. 1b). In Fig. 2a) the object is a task to be executed on a machine A together with a plan for the execution procedure. As before, they can move to machine B. Finally, in Fig. 2b) a workflow is with an employee A, moving to employee B afterwards. Note, that in all of these examples, also the internal state of the object is changed when

2015


MAP
L
Manufacturing Automation
Planning Lab - UFU

- RFID System: Automatic Identification Data Capture
  - Components
    - Tag
    - Reader/ Antenna
    - Middleware
  - Standard Application
    - Key-value paradigm
    - Several types of identification: Product (GTIN), Place (GLN), Shipping (SSCC), Assets (GRAI, GIAI) and Services (GSRN)

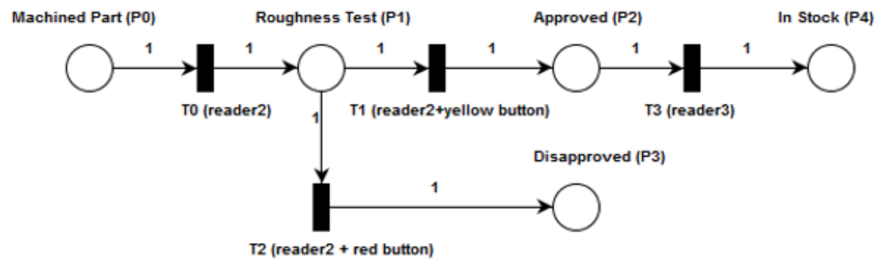


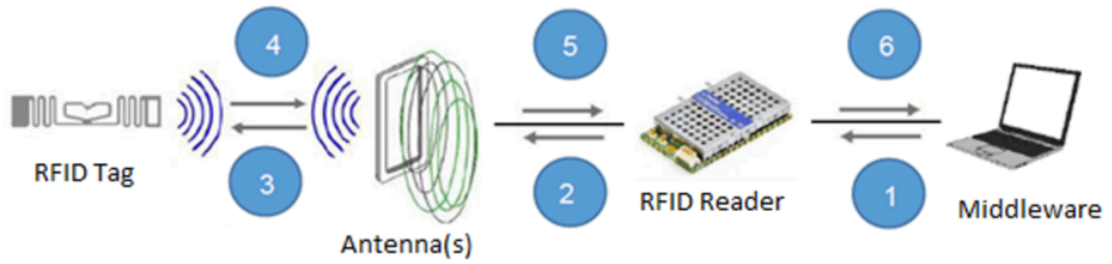


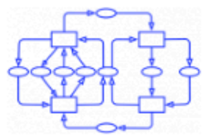
- RFID + Petri Net
  - Generates a contingent decentralized system
  - Introduces process on operational level for each tagged object
  - Supports both descriptive and normative process
  - Integrates design* and implementation with the same mathematical based

* Discrete Event System

RFID
Process (Petri Net)
Value Generati on



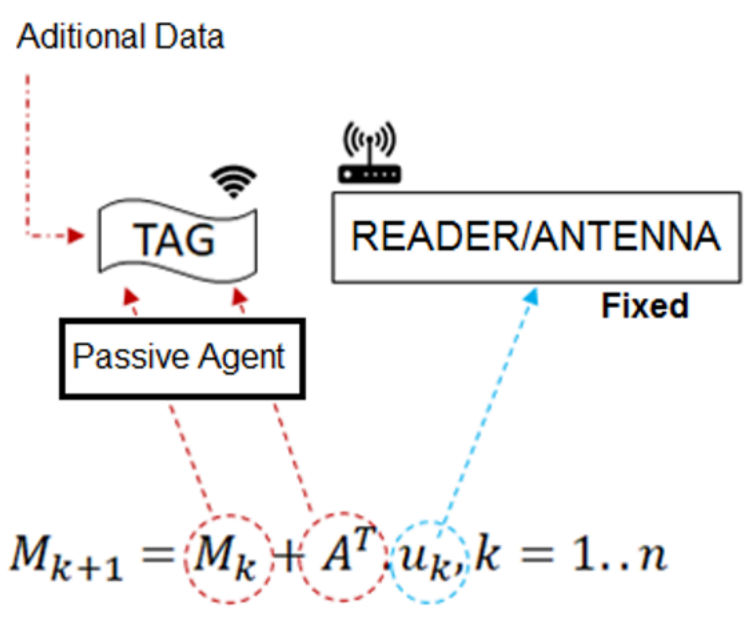




UFU PNRD and iPNRD

Petri Net Inside RFID Database

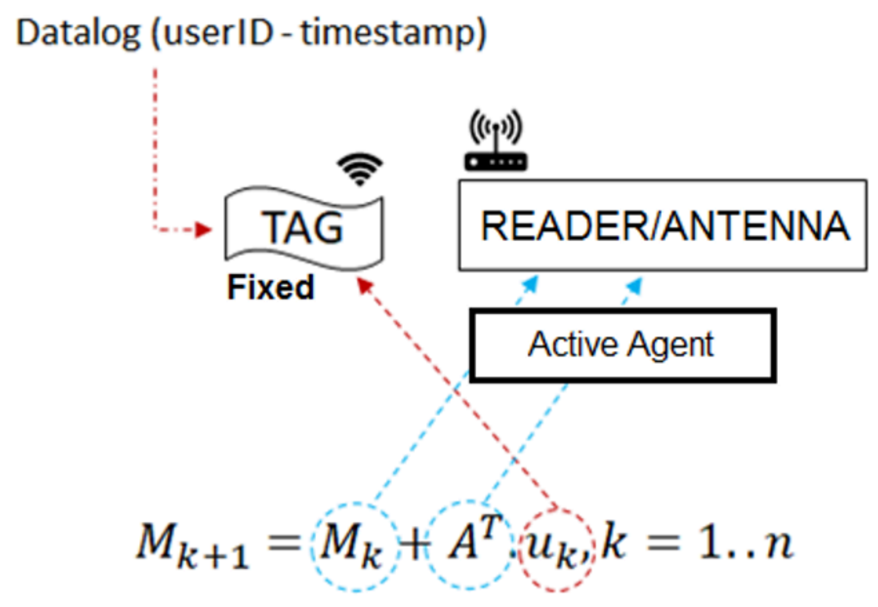
PNRD



$$M_{k+1} = M_k + A^T u_k, k = 1..n$$

Inverted Petri Net Inside RFID Database

iPNRD



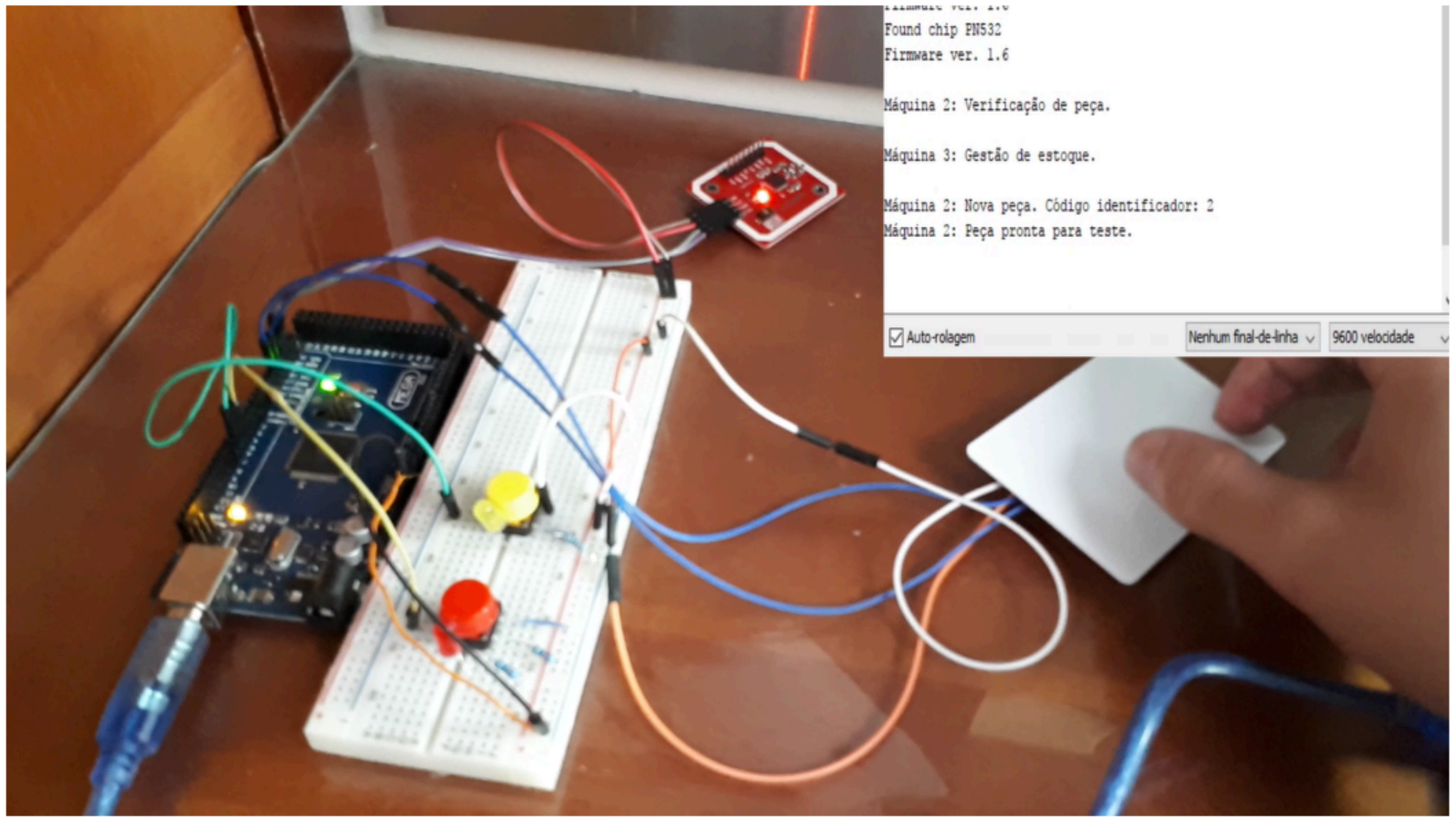
$$M_{k+1} = M_k + A^T u_k, k = 1..n$$

PNRD/ iPNRD distributes PN next state calculus componentes in RFID devices

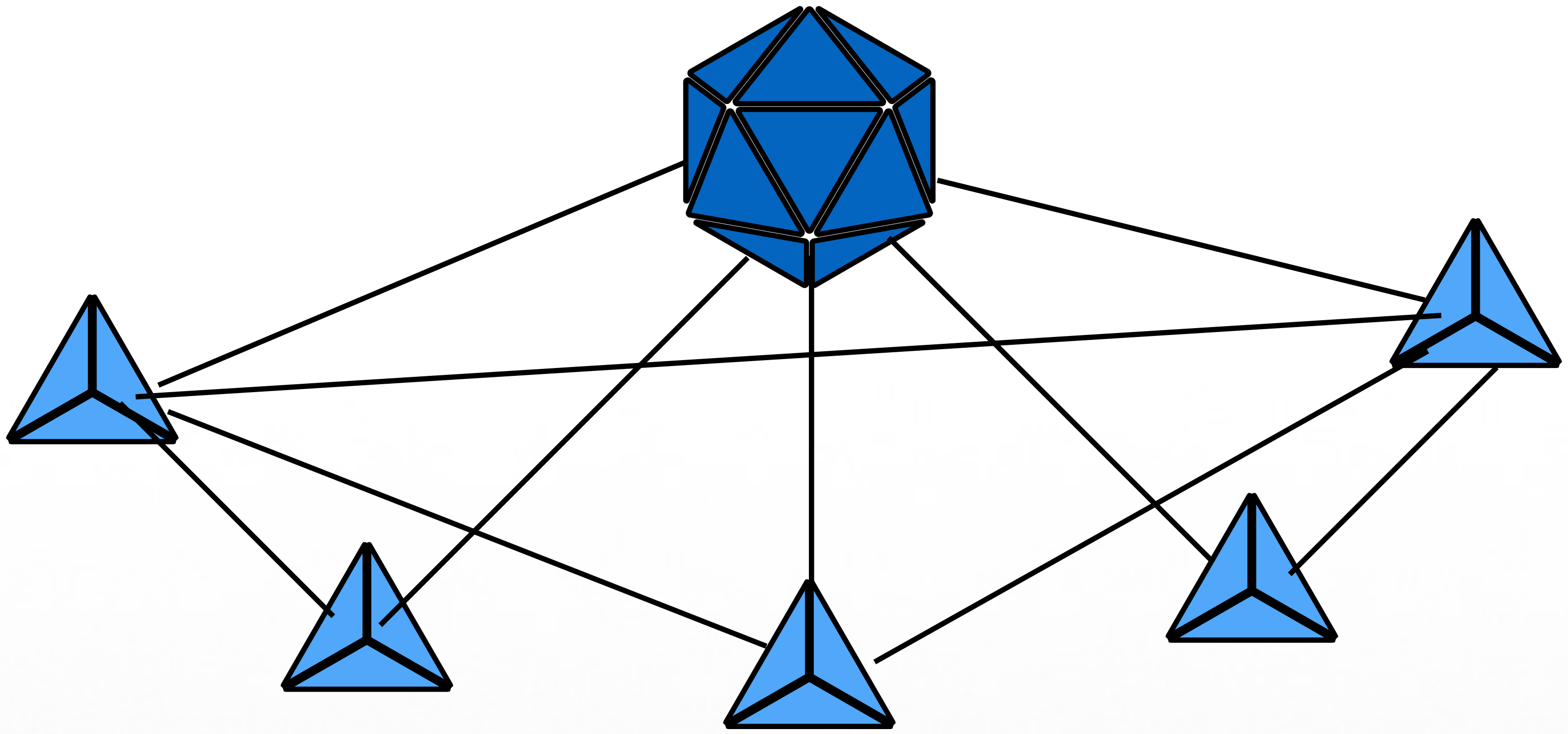
8

Tests

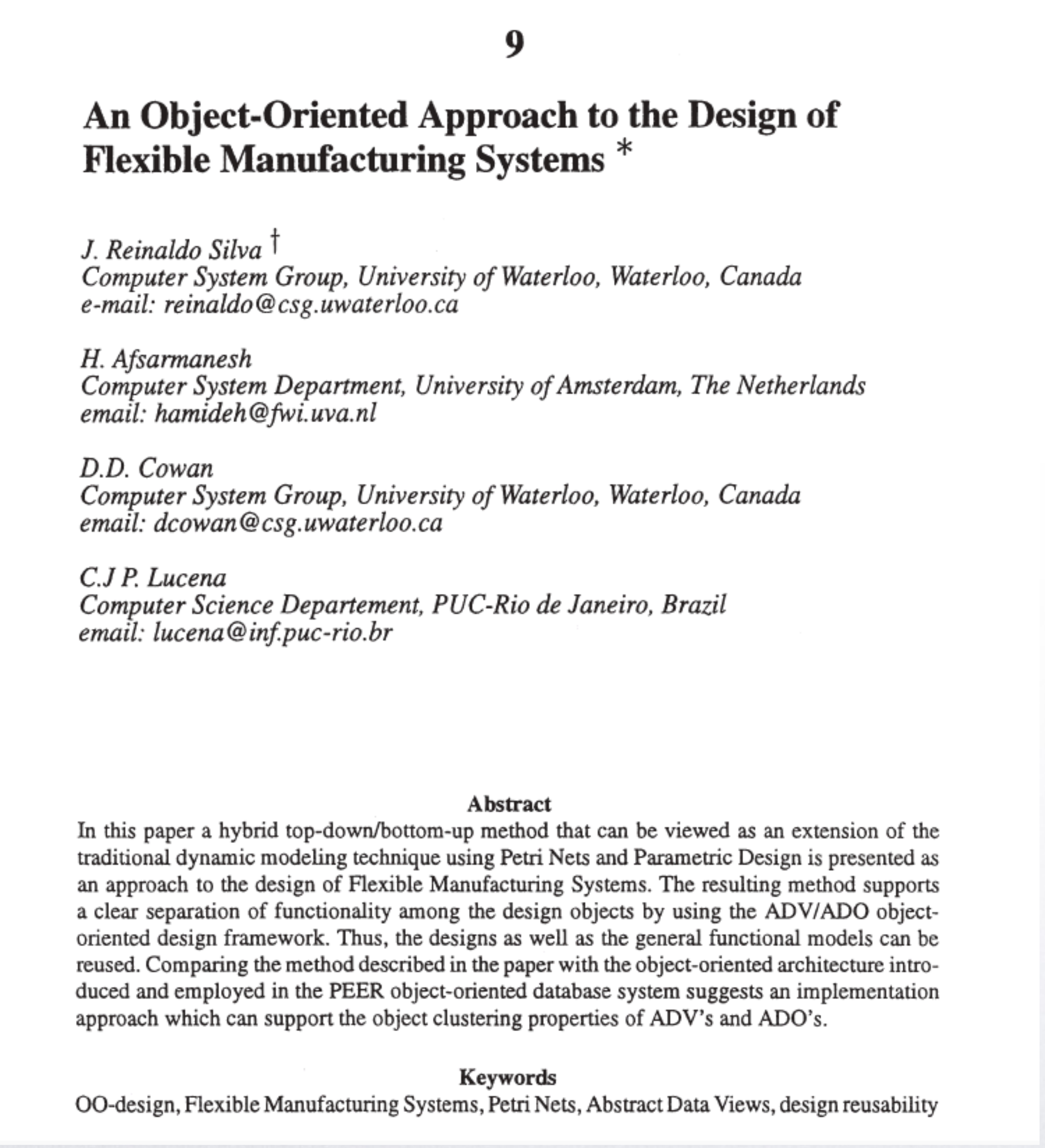
- PNRD test



```
Found chip PN532
Firmware ver. 1.6

Máquina 2: Verificação de peça.

Máquina 3: Gestão de estoque.

Máquina 2: Nova peça. Código identificador: 2
Máquina 2: Peça pronta para teste.
```



From the complete net model one can derive the protocol, which is often too complicated to be of any value. However, after abstracting from internal communication, retaining only a few essential actions, the system often has to satisfy a simple protocol. The notion of branching bisimilarity can be used to verify that the complete system obeys this protocol.



9

An Object-Oriented Approach to the Design of Flexible Manufacturing Systems *

J. Reinaldo Silva †
Computer System Group, University of Waterloo, Waterloo, Canada
e-mail: reinaldo@csg.uwaterloo.ca

H. Afsarmanesh
Computer System Department, University of Amsterdam, The Netherlands
email: hamideh@fwi.uva.nl

D.D. Cowan
Computer System Group, University of Waterloo, Waterloo, Canada
email: dcowan@csg.uwaterloo.ca

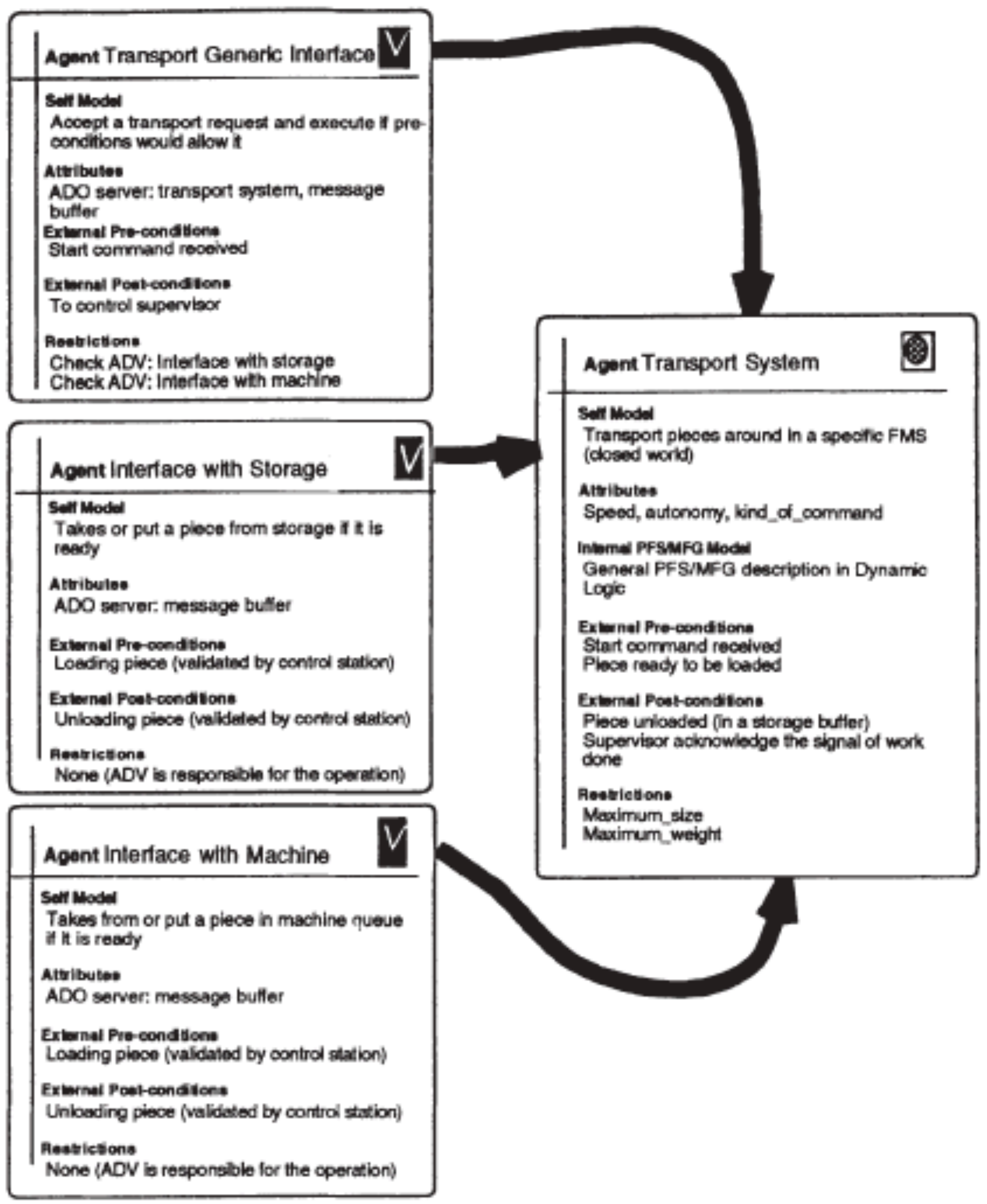
C.J P. Lucena
Computer Science Departement, PUC-Rio de Janeiro, Brazil
email: lucena@inf.puc-rio.br

Abstract

In this paper a hybrid top-down/bottom-up method that can be viewed as an extension of the traditional dynamic modeling technique using Petri Nets and Parametric Design is presented as an approach to the design of Flexible Manufacturing Systems. The resulting method supports a clear separation of functionality among the design objects by using the ADV/ADO object-oriented design framework. Thus, the designs as well as the general functional models can be reused. Comparing the method described in the paper with the object-oriented architecture introduced and employed in the PEER object-oriented database system suggests an implementation approach which can support the object clustering properties of ADV's and ADO's.

Keywords

OO-design, Flexible Manufacturing Systems, Petri Nets, Abstract Data Views, design reusability



Agent Transport Generic Interface

Self Model
Accept a transport request and execute if pre-conditions would allow it

Attributes
ADO server: transport system, message buffer

External Pre-conditions
Start command received

External Post-conditions
To control supervisor

Restrictions
Check ADV: Interface with storage
Check ADV: Interface with machine

Agent Interface with Storage

Self Model
Takes or put a piece from storage if it is ready

Attributes
ADO server: message buffer

External Pre-conditions
Loading piece (validated by control station)

External Post-conditions
Unloading piece (validated by control station)

Restrictions
None (ADV is responsible for the operation)

Agent Interface with Machine

Self Model
Takes from or put a piece in machine queue if it is ready

Attributes
ADO server: message buffer

External Pre-conditions
Loading piece (validated by control station)

External Post-conditions
Unloading piece (validated by control station)

Restrictions
None (ADV is responsible for the operation)

Agent Transport System

Self Model
Transport pieces around in a specific FMS (closed world)

Attributes
Speed, autonomy, kind_of_command

Internal PFS/MFG Model
General PFS/MFG description in Dynamic Logic

External Pre-conditions
Start command received
Piece ready to be loaded

External Post-conditions
Piece unloaded (in a storage buffer)
Supervisor acknowledge the signal of work done

Restrictions
Maximum_size
Maximum_weight

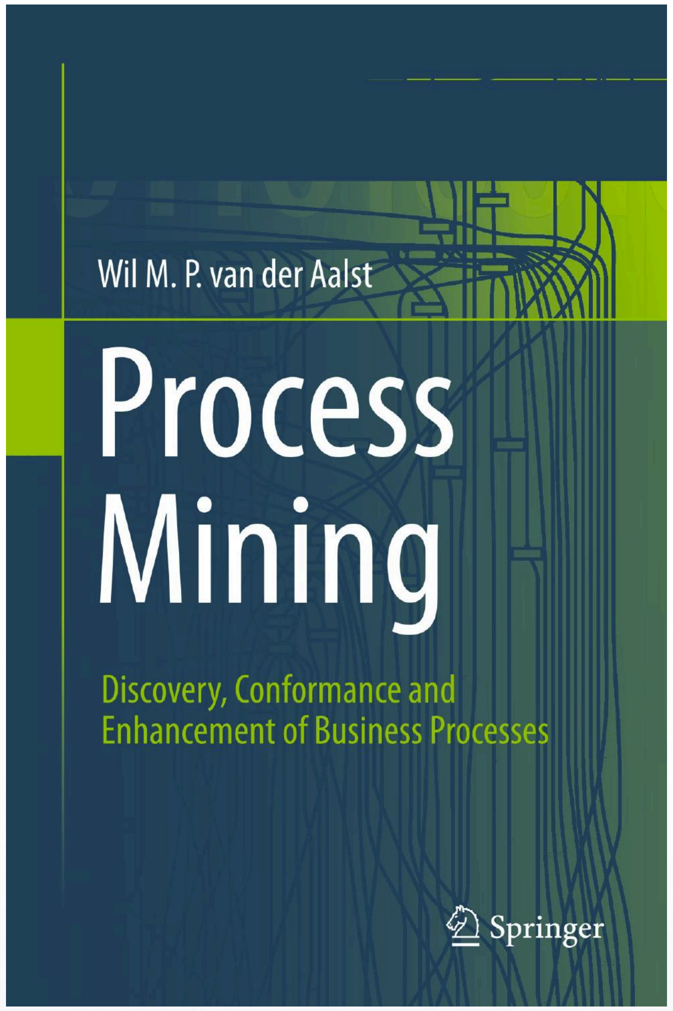
IFIP-Balanced Automation Systems, 1995

Decomposing a system (stepwise) means defining subsystems (eventually independent systems) and (first) their interface (with the main system or with any other system).

➡ Object-oriented Systems Design

Difficulties with OO-PN

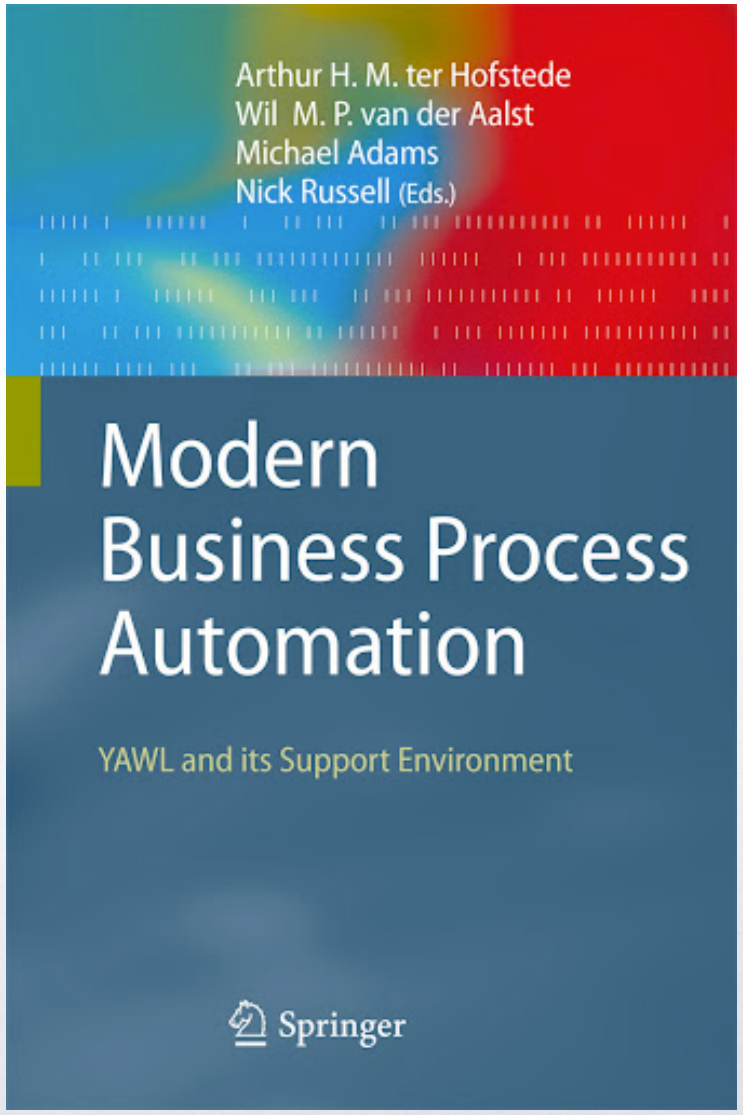
Wil van der Aalst


Wil M. P. van der Aalst

Process Mining

Discovery, Conformance and Enhancement of Business Processes

Springer

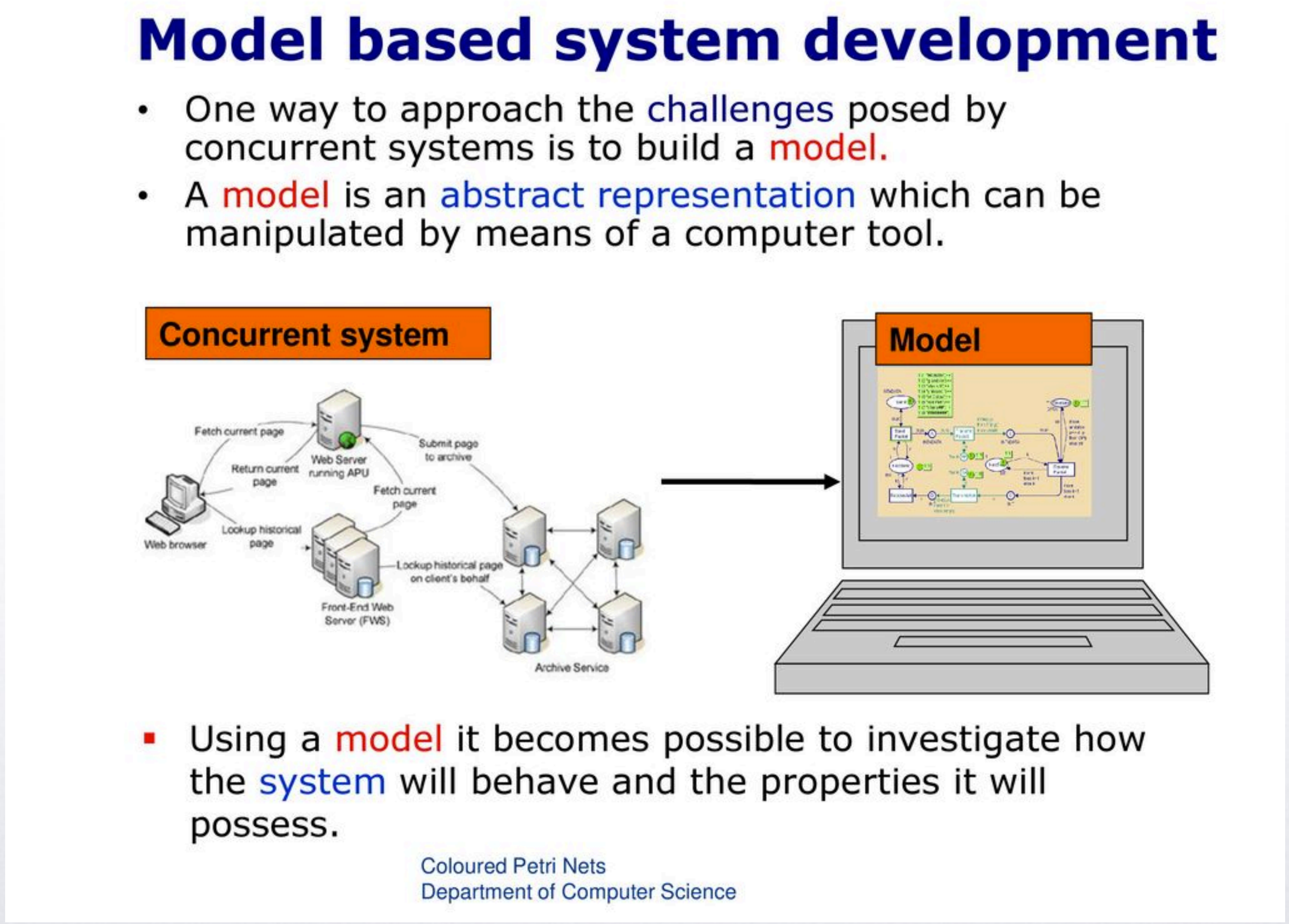

Arthur H. M. ter Hofstede
Wil M. P. van der Aalst
Michael Adams
Nick Russell (Eds.)

Modern Business Process Automation

YAWL and its Support Environment

Springer

Old and new challenges to System Modeling (with Petri nets)



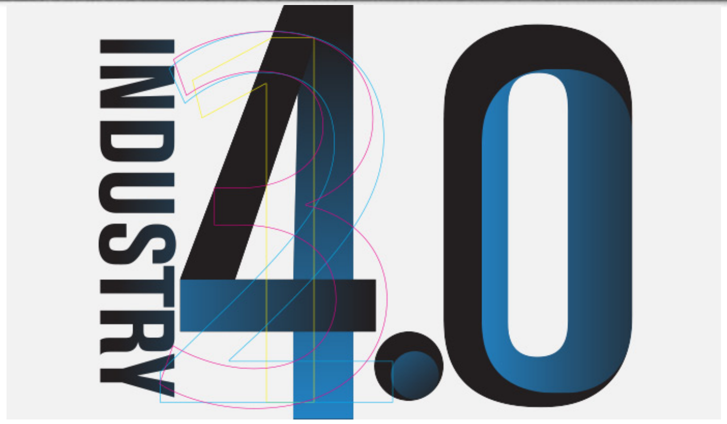
Model based system development

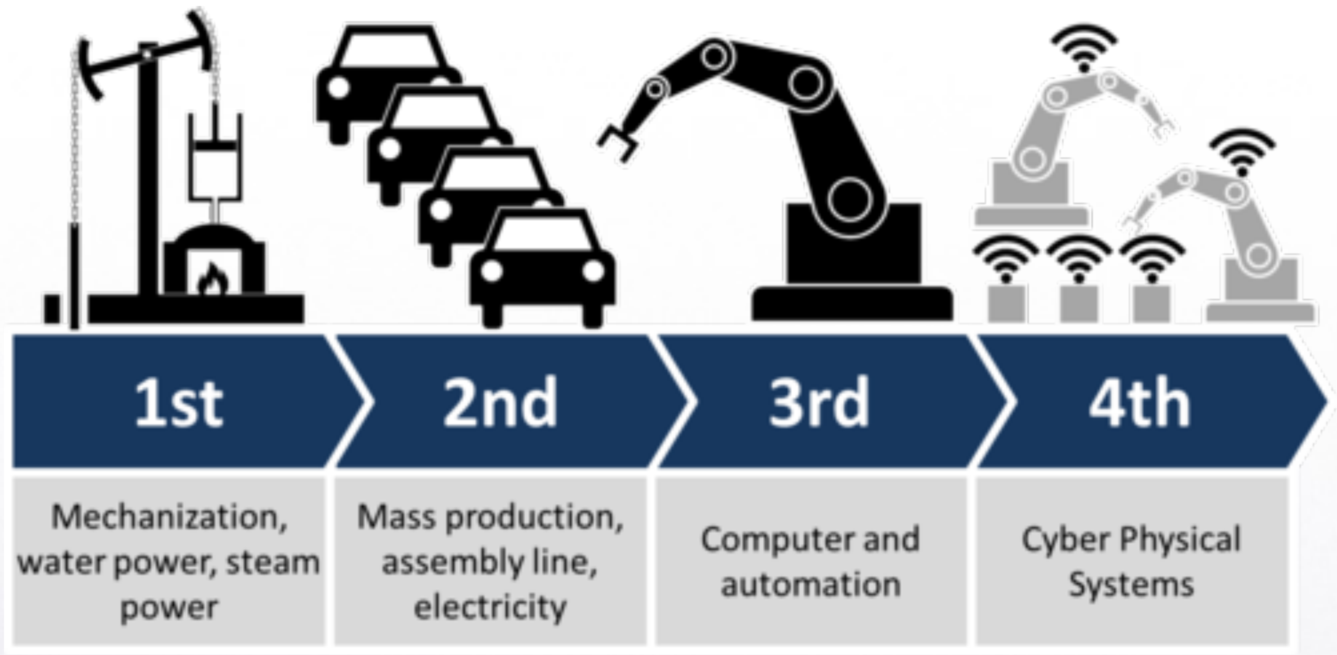
- One way to approach the challenges posed by concurrent systems is to build a model.
- A model is an abstract representation which can be manipulated by means of a computer tool.

Concurrent system
Fetch current page
Return current page
Web Server running APU
Submit page to archive
Fetch current page
Lookup historical page
Web browser
Front-End Web Server (FWS)
Lookup historical page on client's behalf
Archive Service
Model

- Using a model it becomes possible to investigate how the system will behave and the properties it will possess.

Coloured Petri Nets
Department of Computer Science


INDUSTRY
4.0

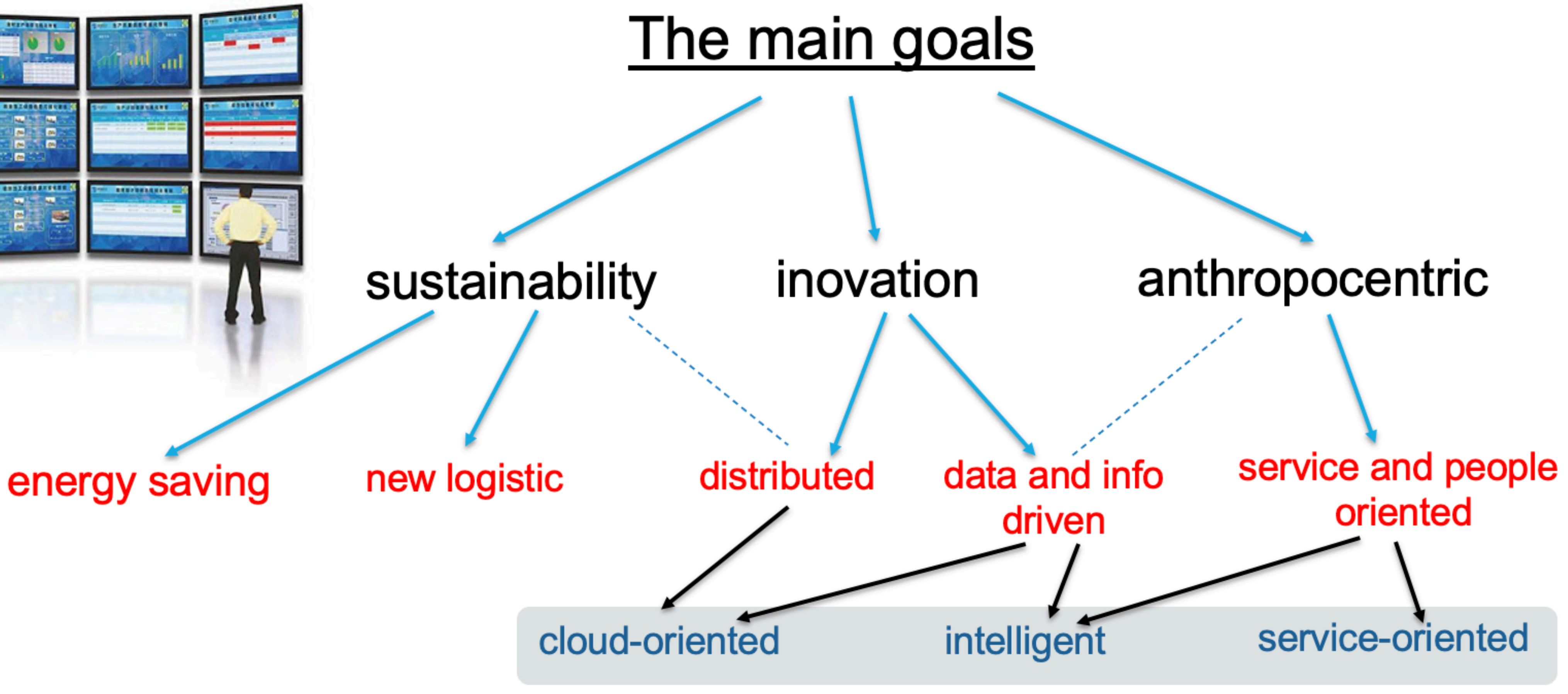

1st
Mechanization, water power, steam power
2nd
Mass production, assembly line, electricity
3rd
Computer and automation
4th
Cyber Physical Systems

Towards a Formal Design to Service-Oriented Cloud Manufacturing

José Reinaldo Silva
D-Lab, Escola Politécnica
Universidade de São Paulo

23-26/nov/2020

The new challenges to intelligent manufacturing



The new tendencies: cloud and service manufacturing

Tendencies

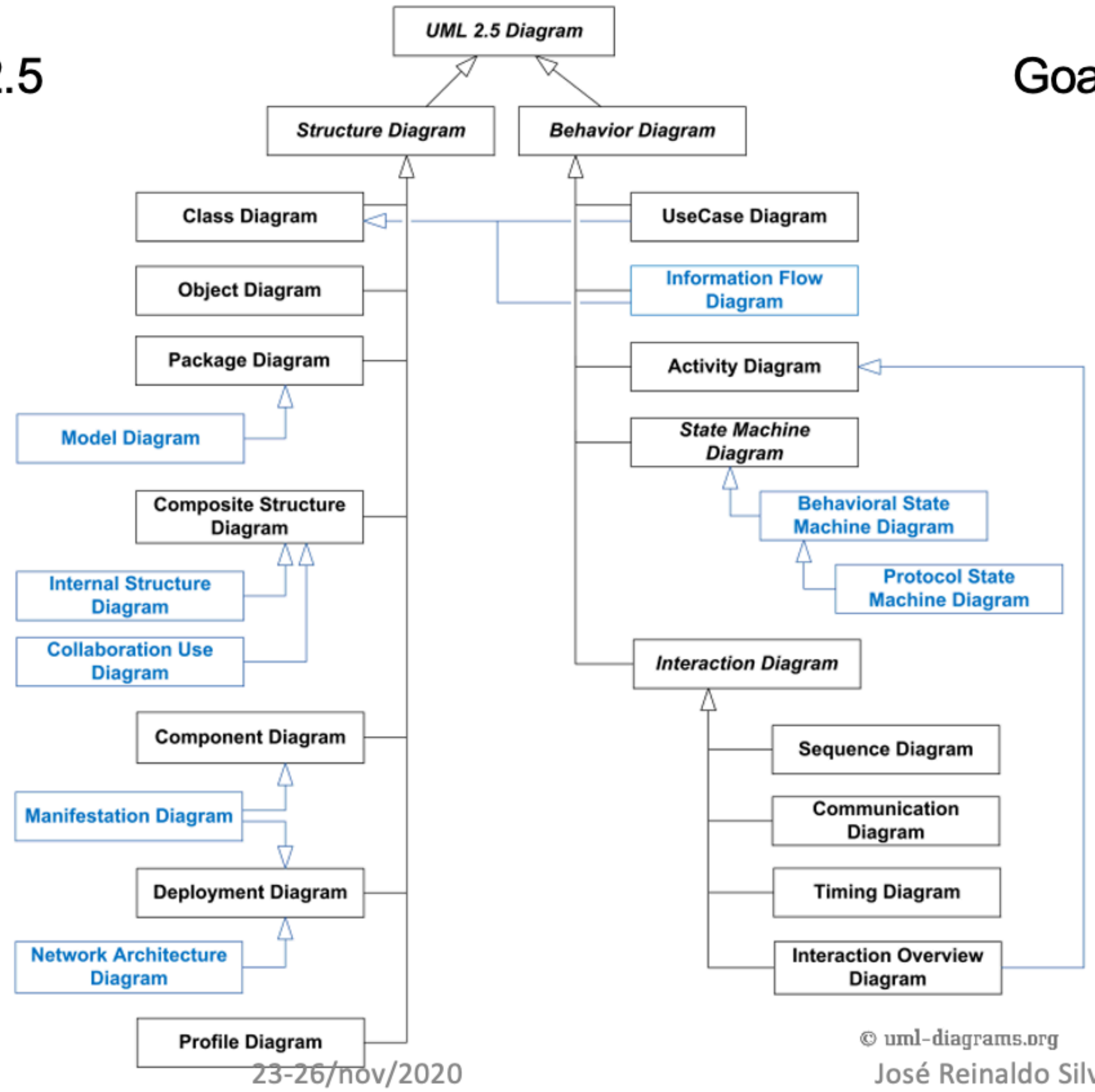
Intelligent and e-manufacturing, IoT,digital twins, etc.

CMfg

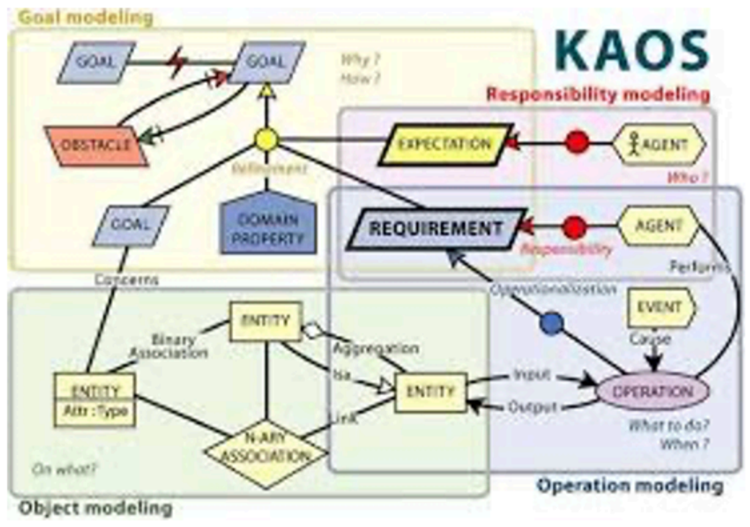
Service Mfg

The design challenge and the proposed method

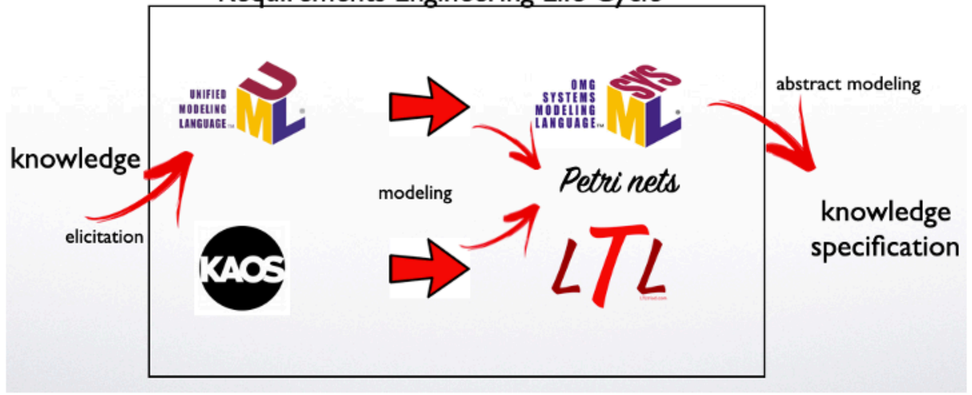
UML 2.5



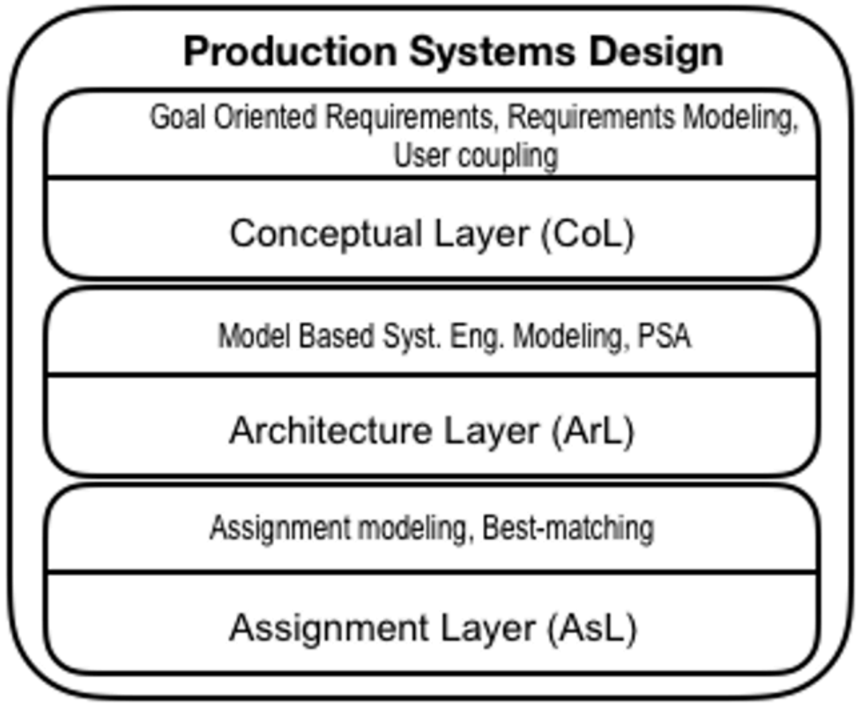
Goal-oriented requiments Engineering (GORE)



Requirements Engineering Life Cycle

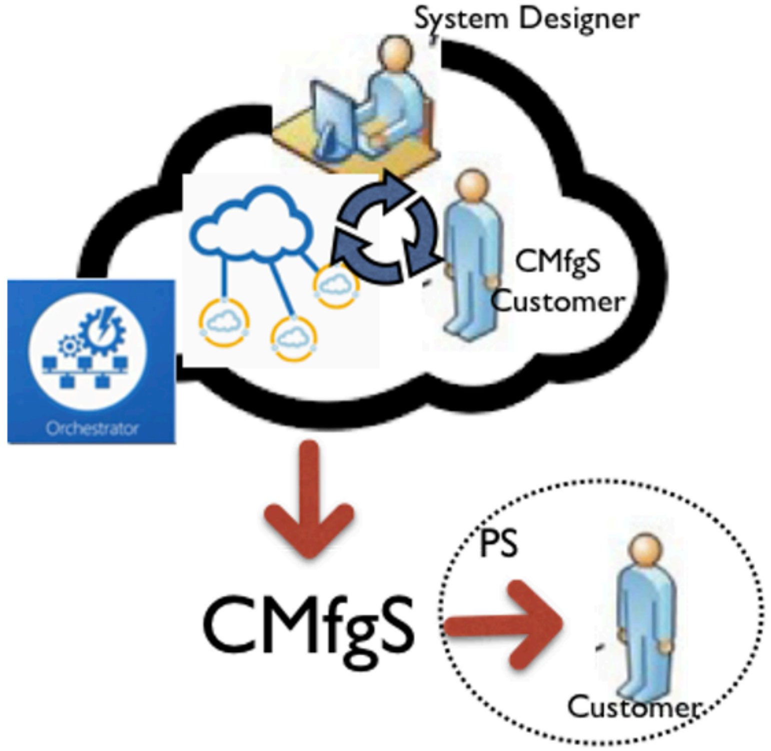


Reinforcing dynamic design: design becomes a permanent activity



Production Systems Design

Goal Oriented Requirements, Requirements Modeling, User coupling
Conceptual Layer (CoL)
Model Based Syst. Eng. Modeling, PSA
Architecture Layer (ArL)
Assignment modeling, Best-matching
Assignment Layer (AsL)



Terminamos aqui a nossa disciplina e agradeço a todos pela participação.