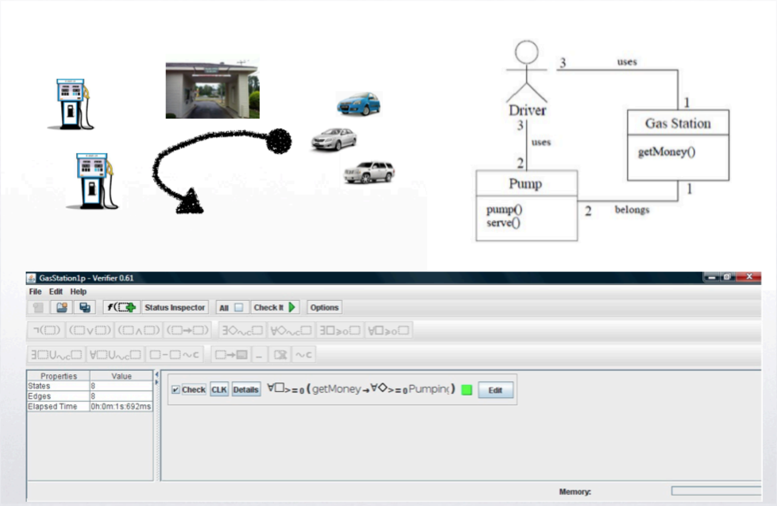
PMR 5237

Modelagem e Design de Sistemas Discretos em Redes de Petri

Aula 10: Modelagem formal de sistemas baseada no espaço de estados



Prof. José Reinaldo Silva

Escola Politécnica da USP

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

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).

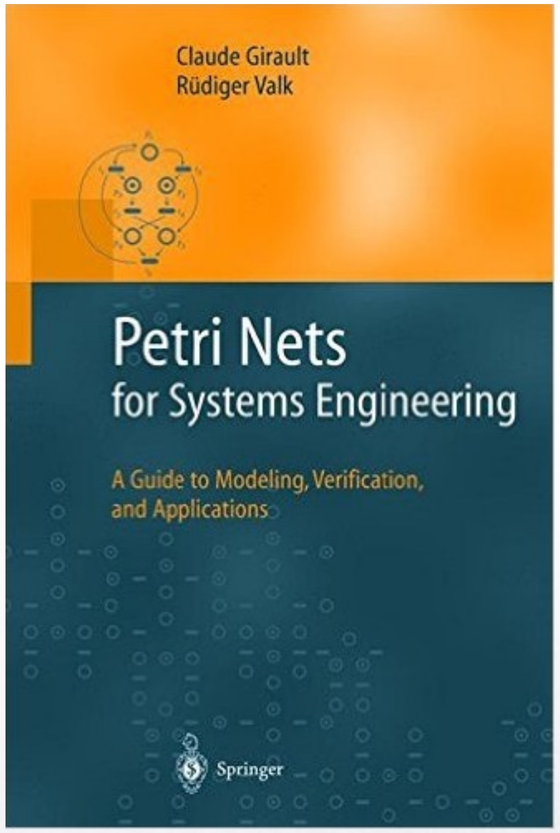
Early intervention programs, such as Head Start, have the potential to alleviate the risks of poverty associated with poor school adjustment and achievement (V. E. Lee & Burkam, 2002; Shonkoff & [illegible]llips, 2000). Research ha[illegible]shown that such interventions are most successful when they 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).

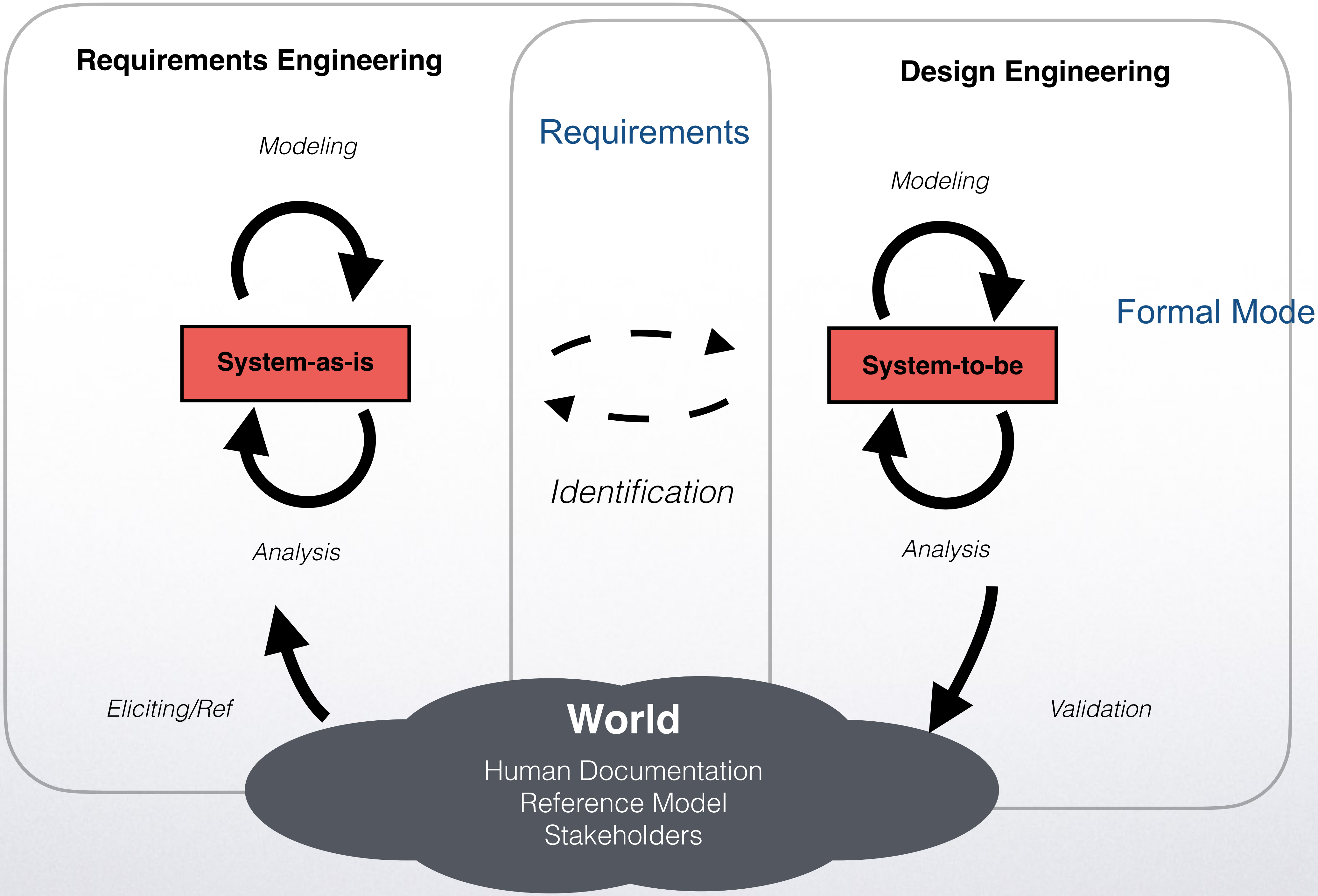
Developmental theory and research suggest that the preschool classroom is a naturally occurring context for peer play through which [illegible] acquire knowledge and skil[illegible] (Coplan & Arbeau, 2009; [illegible] G. Singer, Golinkoff, & Hirsh-Pa[illegible], 2006). A growing body [illegible] [illegible] mathematics skills in preschool, [illegible] 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 [illegible] academic

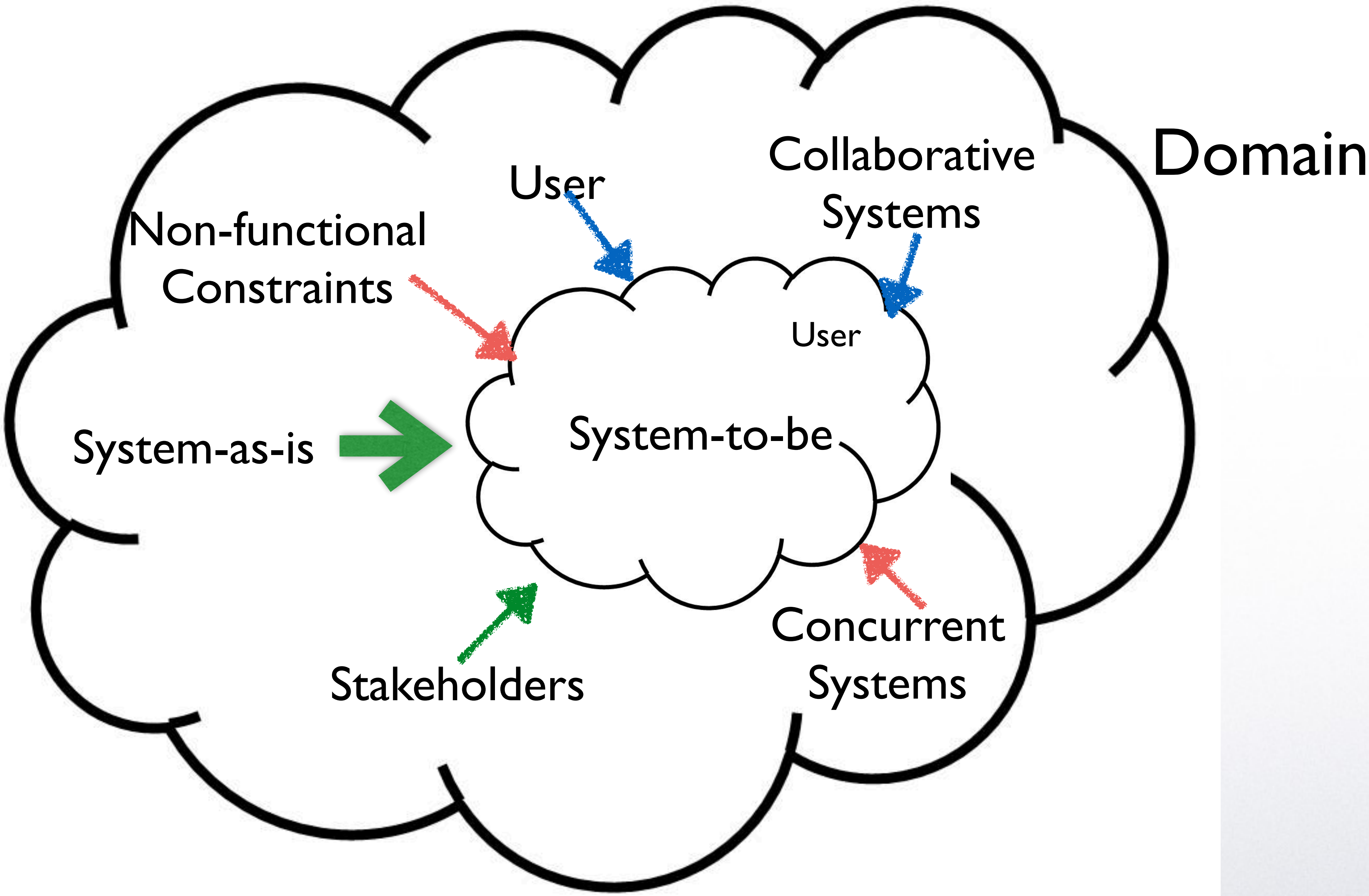
Elizabeth R. Bell, Daryl B. Greenfield, Rebecc[illegible] [illegible]ky-Shearer, and Tracy M. Carter, Department of Psychology, [illegible] of Miami.
Correspondence concerning this article should be addressed to Elizabeth R. Bell, who is now at Accelerate Learning Inc., 5700 Richmond Avenue, Suite 1025, Houston, TX 77056. E-mail: ebell@acceleratelearning.co[illegible]

Artigo Final: teremos mais um milestone na aula que vem, onde vocês devem inserir no artigo mais detalhes de como vão usar redes de Petri no tema proposto. O deadline será na sexta 3/12. Teremos mais uma aula e outro milestone para o dia 14/12 onde será apresentado os detalhes do uso das redes de Petri. O artigo final fica para o dia 21/12.

Part II. Modelling




Requirements Engineering
Modeling
System-as-is
Analysis
Eliciting/Ref
Requirements
Identification
Design Engineering
Modeling
Formal Model
System-to-be
Analysis
Validation
World
Human Documentation
Reference Model
Stakeholders


Domain
User
Collaborative
Systems
Non-functional
Constraints
User
System-as-is
System-to-be
Stakeholders
Concurrent
Systems

Na aula passada vimos as técnicas de design (as principais) para a modelagem de sistemas em redes de Petri, nesta aula trataremos brevemente da técnica de building blocks aplicada às redes de alto nível e seguiremos na discussão dos métodos de modelagem de sistemas usando Redes de Petri, hoje com o método orientado a estados .

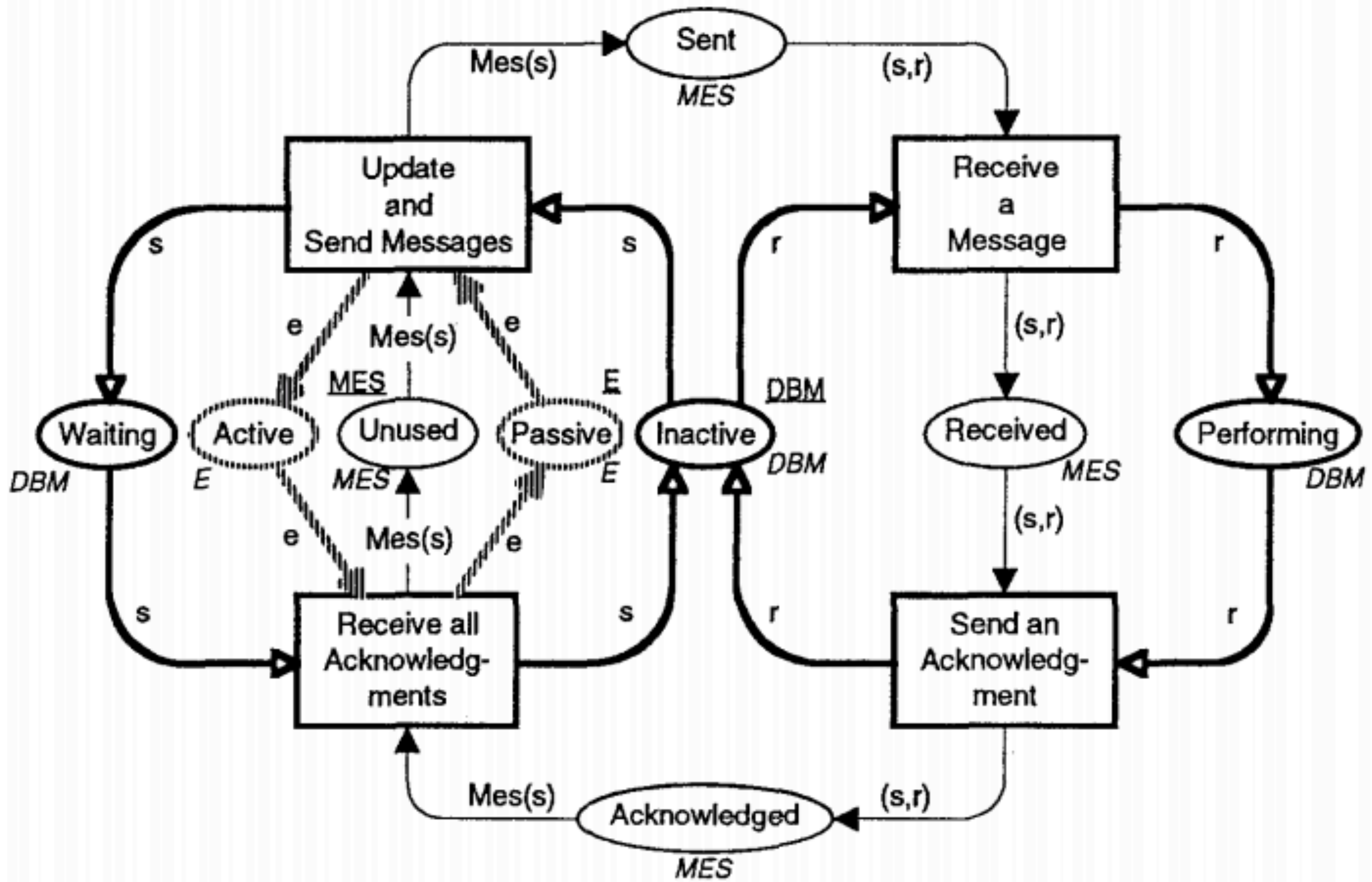
PN Basic Properties

1) *boundedness*, characterising finiteness of the state space.
2) *liveness*, related to potential fireability in all reachable markings. *Deadlock-freeness* is a weaker condition in which global infinite activity (i.e. fireability) of the net system model is guaranteed, but some parts of it may not work at all.
3) *reversibility*, characterising recoverability of the initial marking from any reachable marking.
4) *mutual exclusion*, dealing with the impossibility of simultaneous *submarkings* (p-mutex) or *firing concurrency* (t-mutex).

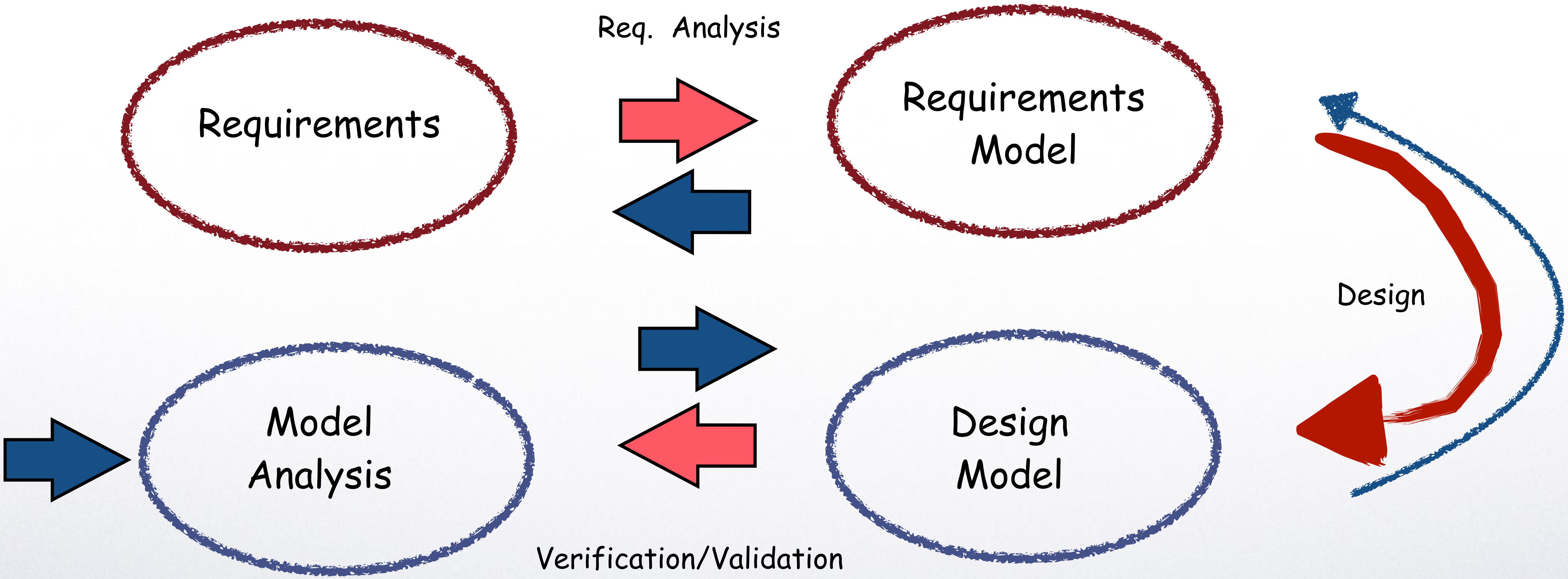
Fairness properties

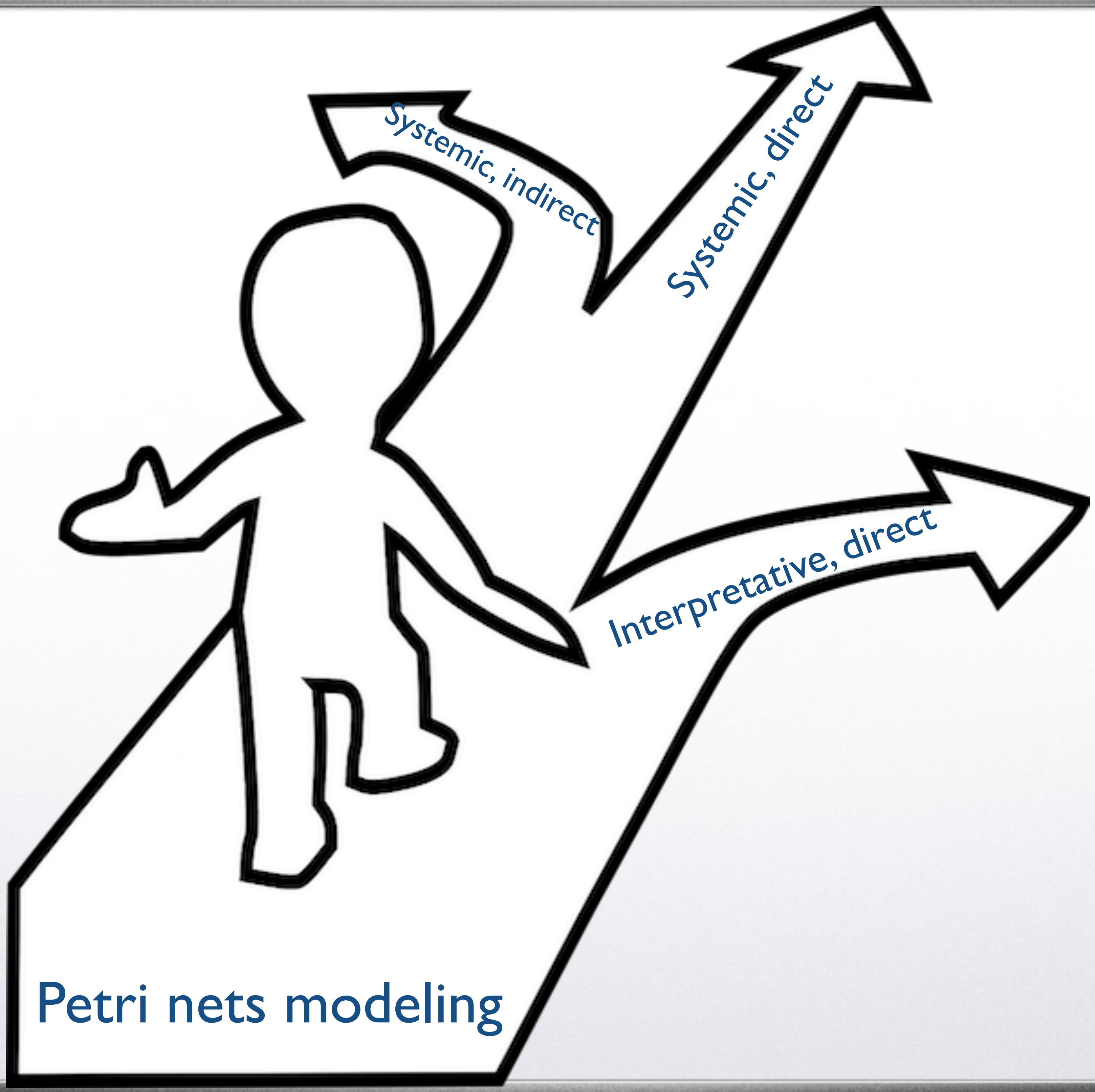
- Are only relevant if there are Infinite Firing Sequences (IFS), otherwise CPN Tools reports: "no infinite occurrence sequences".
- Given a transition t it is often desirable that t appears infinitely often in an IFS.
- Properties reported by CPN Tools
  - t is **impartial**: t occurs infinitely often in every IFS.
  - t is **fair**: t occurs infinitely often in every IFS where t is enabled infinitely often.
  - t is **just**: t occurs infinitely often in every IFS where t is continuously enabled from some point onward
  - **No fairness**: not just, i.e., there is an IFS where t is continuously enabled from some point onward and does not fire anymore

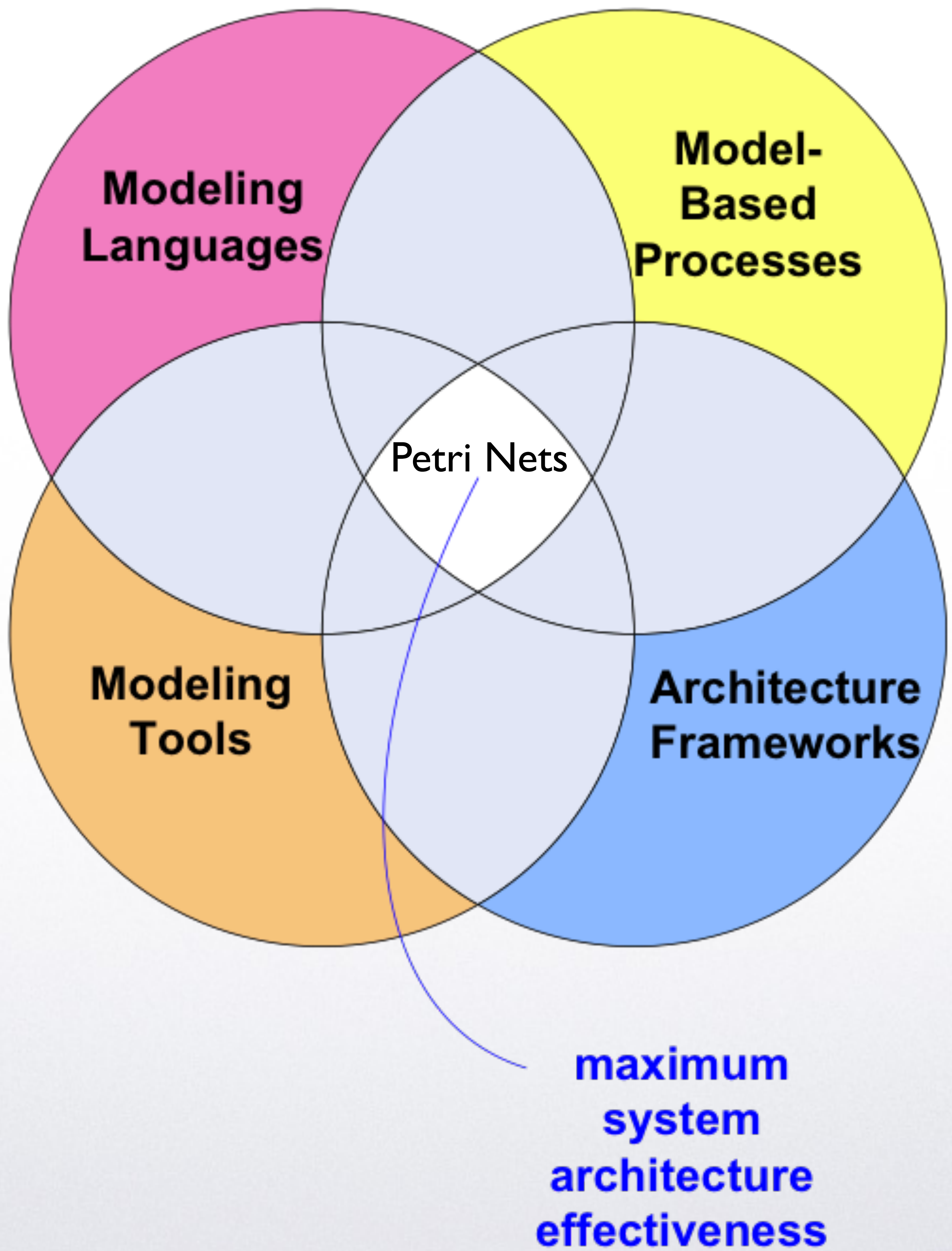
```
val n = 5;
color DBM = index d with 1..n  declare ms;
color PR = product DBM * DBM  declare mult;
fun diff(x,y) = (x<>y);
color MES = subset PR by diff  declare ms;
color E = with e;
fun Mes(s) = mult'PR(1`s,DBM-1`s);
var s, r : DBM;
```



Use of Petri Nets in Design

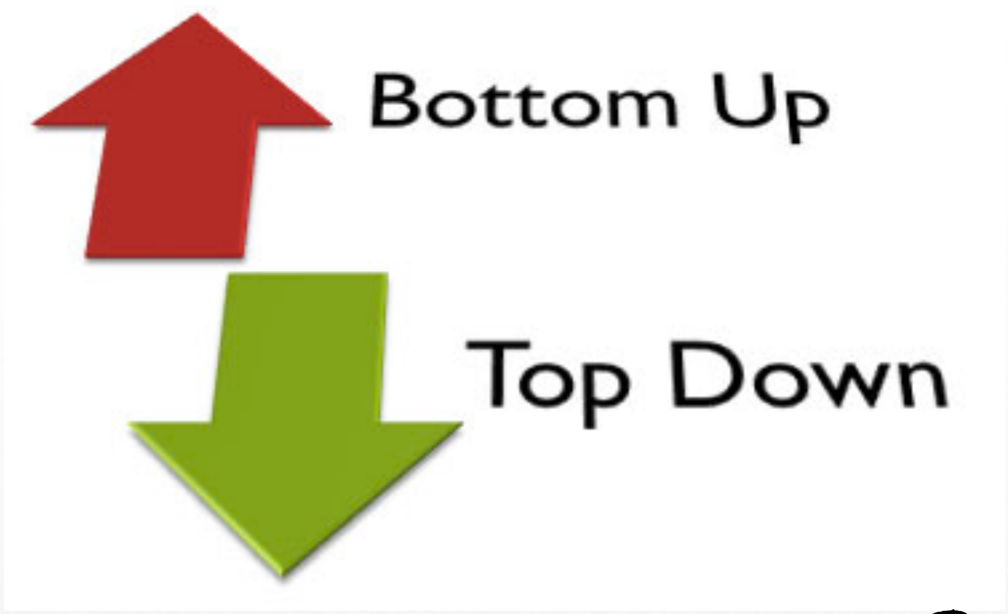



Systemic, indirect
Systemic, direct
Interpretative, direct
Petri nets modeling


Modeling Languages
Model-Based Processes
Petri Nets
Modeling Tools
Architecture Frameworks
maximum system architecture effectiveness

Structured approach

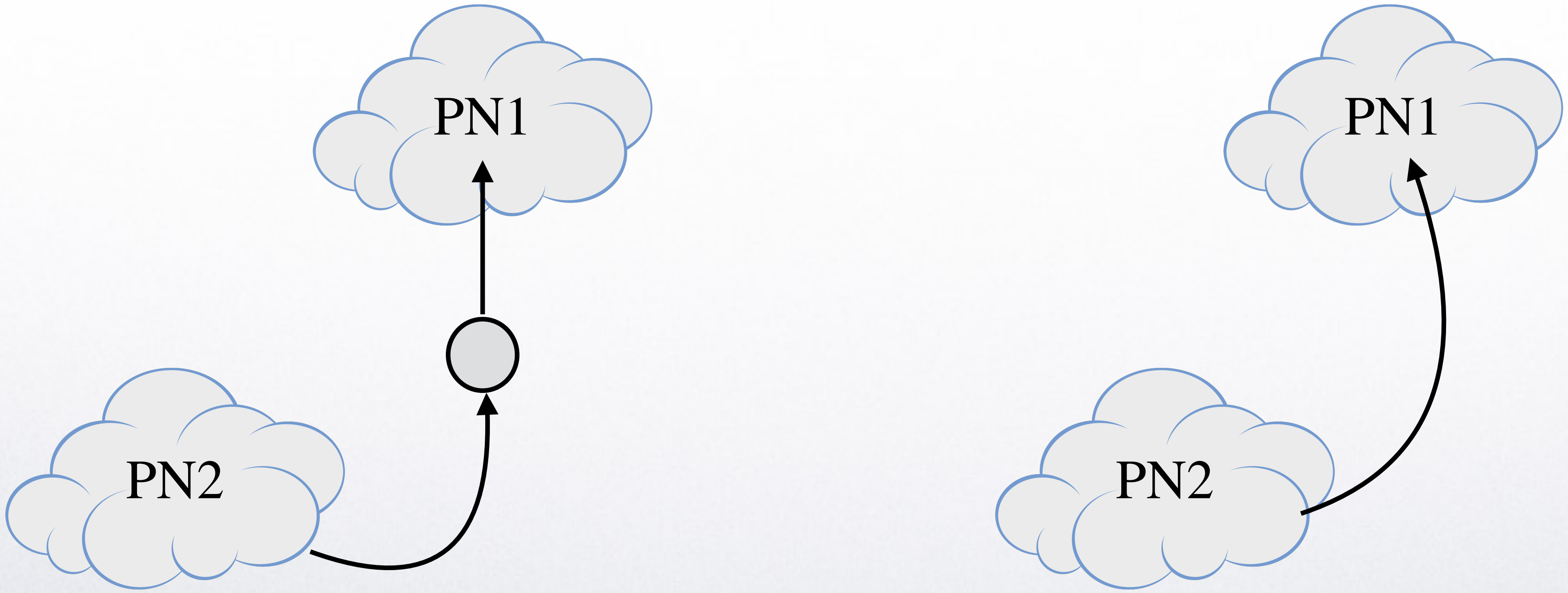
Reusability,
Based on compositionality



Systemic,
Based on refinements

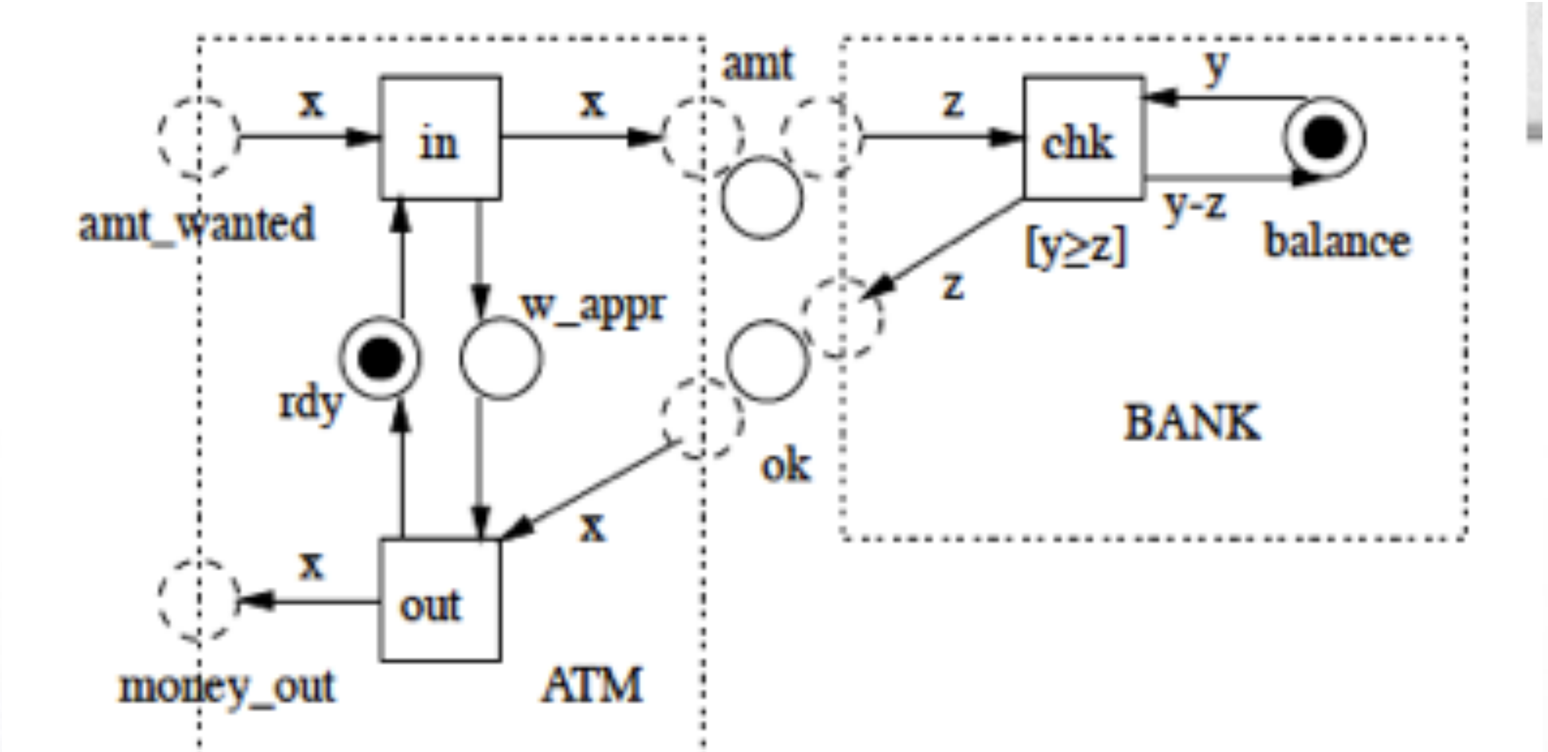
Communication should be synchronous or asynchronous...

... asynchronous communication must be represented by place fusion or arc addition.



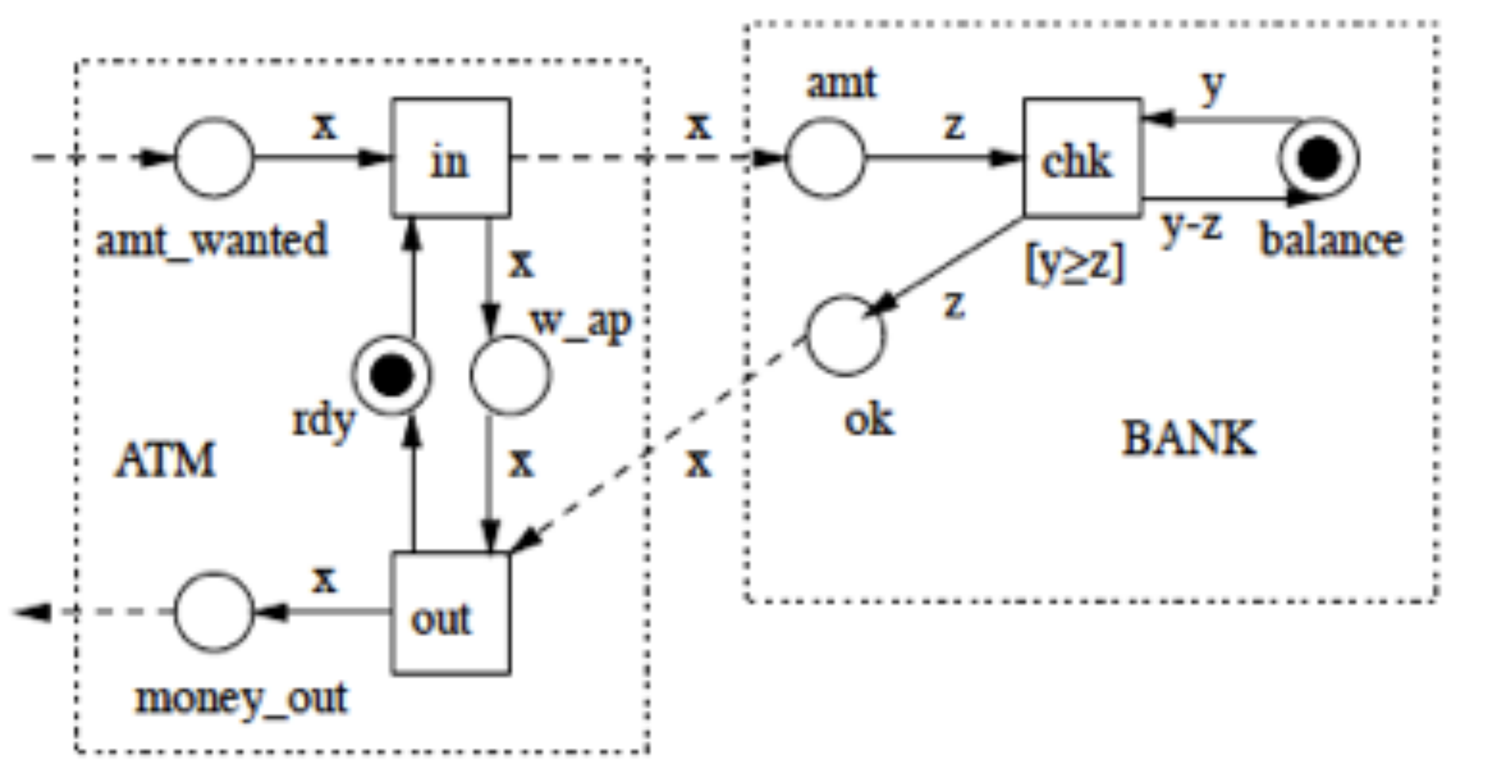
Automatic Teller Machines

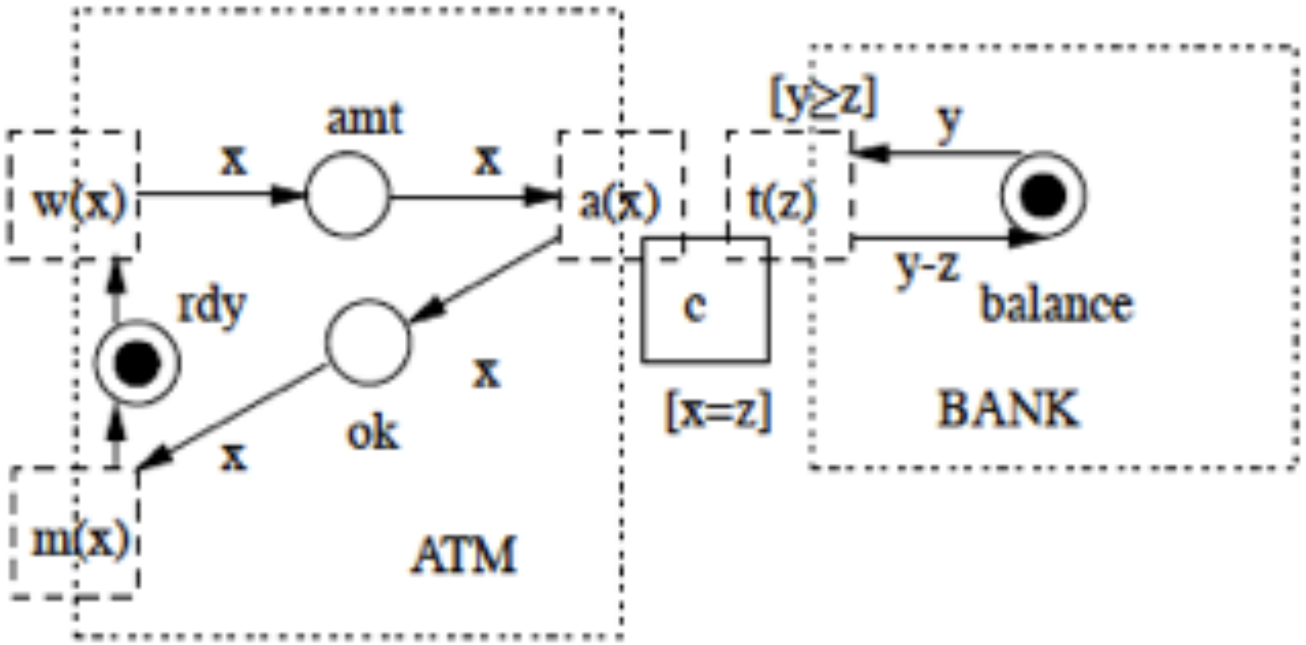
A classic automation example



Applying place fusion to the modeling with CPN requires that the color of marks in the fusion places be respected and preserved.

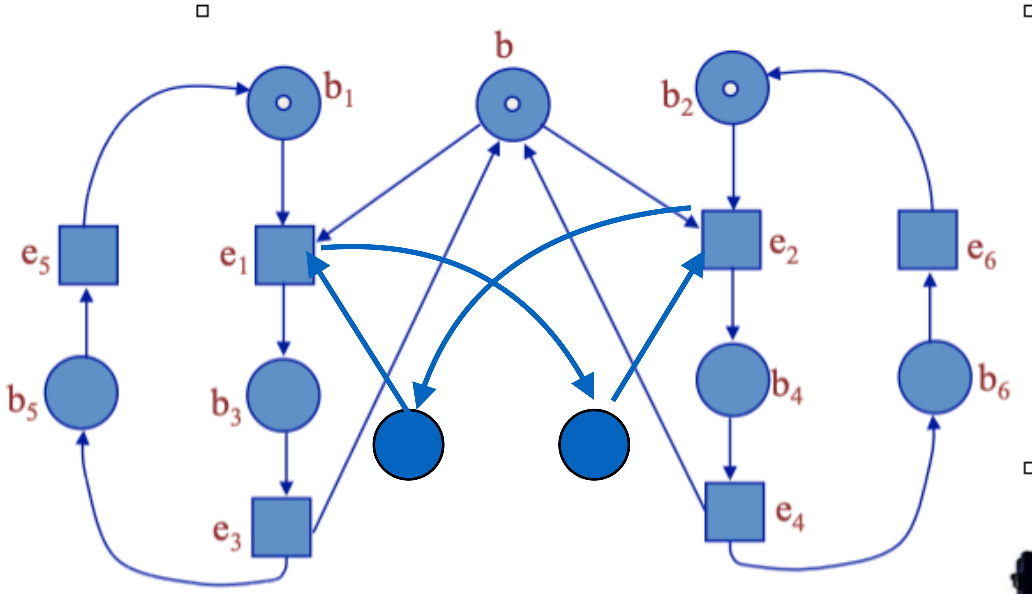
Applying arc addition to the modeling with CPN requires the addition of an arc expression.





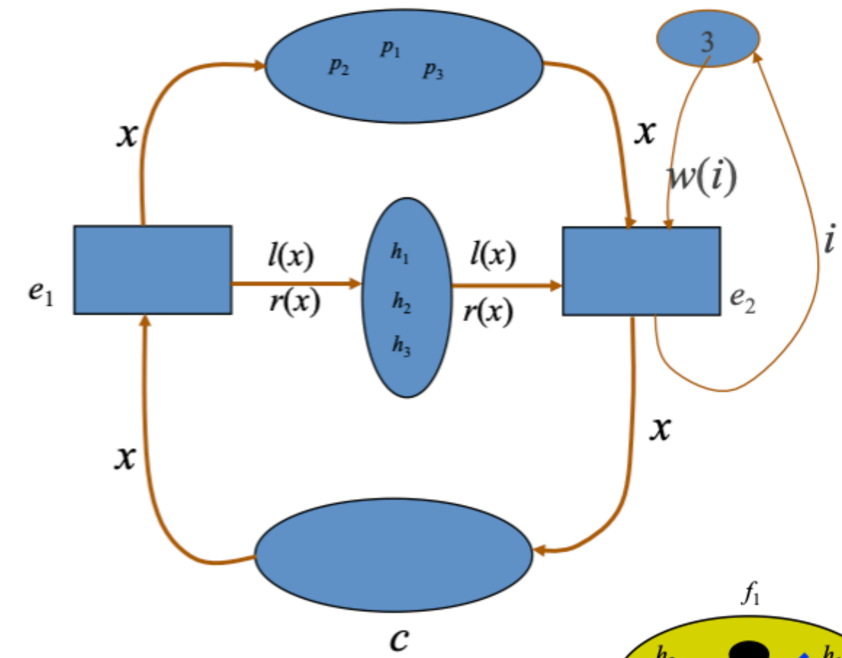
Applying transition fusion to the modeling with CPN requires some filter expressions connected to the transitions and the fusion.

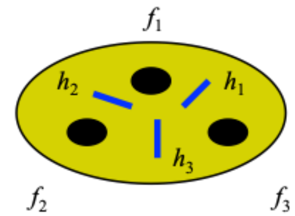
This example illustrates that transition fusion with coloured nets is less straightforward than place fusion. Arc addition is in between the two. Existing tools for modelling and simulating coloured nets, such as Design/CPN ([Jen92b]) and ExSpect ([HSV91], [Bak96]), support only place fusion for this very reason.



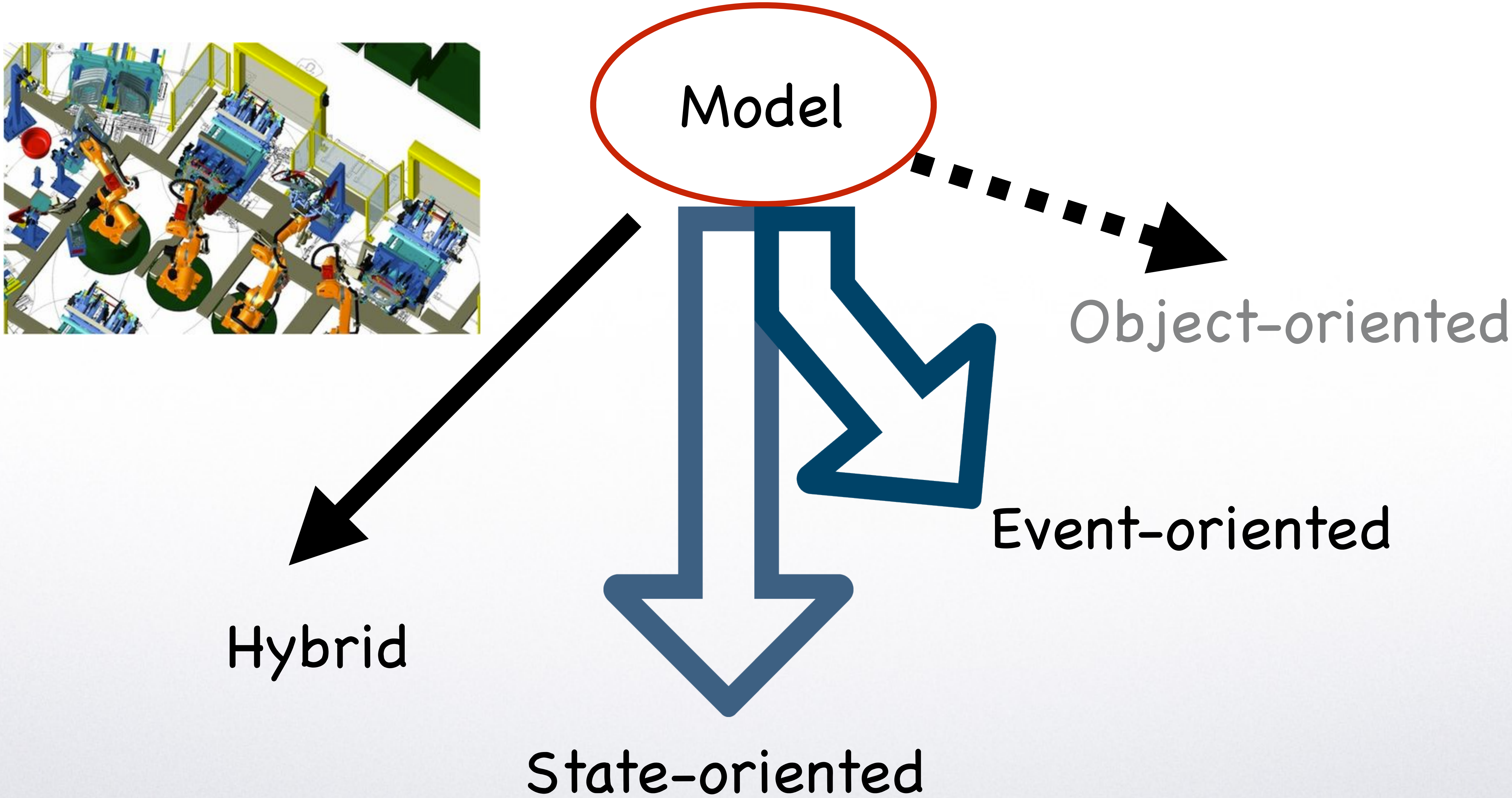
$P=\{p_1, p_2, p_3\}$
$H=\{h_1, h_2, h_3\}$
$U=P \cup H$

$l : P \rightarrow H$
$p_i \rightarrow h_i$
$r : P \rightarrow H$
$p_1 \rightarrow h_2$
$p_2 \rightarrow h_3$
$p_3 \rightarrow h_1$
$w: P \rightarrow I$
$p_j \rightarrow j=(i) \bmod 3+1$





Priority directly addresses conflicting firings for which preferences can be indicated. A less-preferred firing will occur only if more-preferred firings cannot. In nets without colour, preferences are attached to transitions. In coloured nets, these preferences also depend on the bindings (the colours of the tokens to be consumed), so it is for example possible to indicate a preference for the largest token in some place. With parametrised transitions, the transition parameters can also be included in determining preferences.

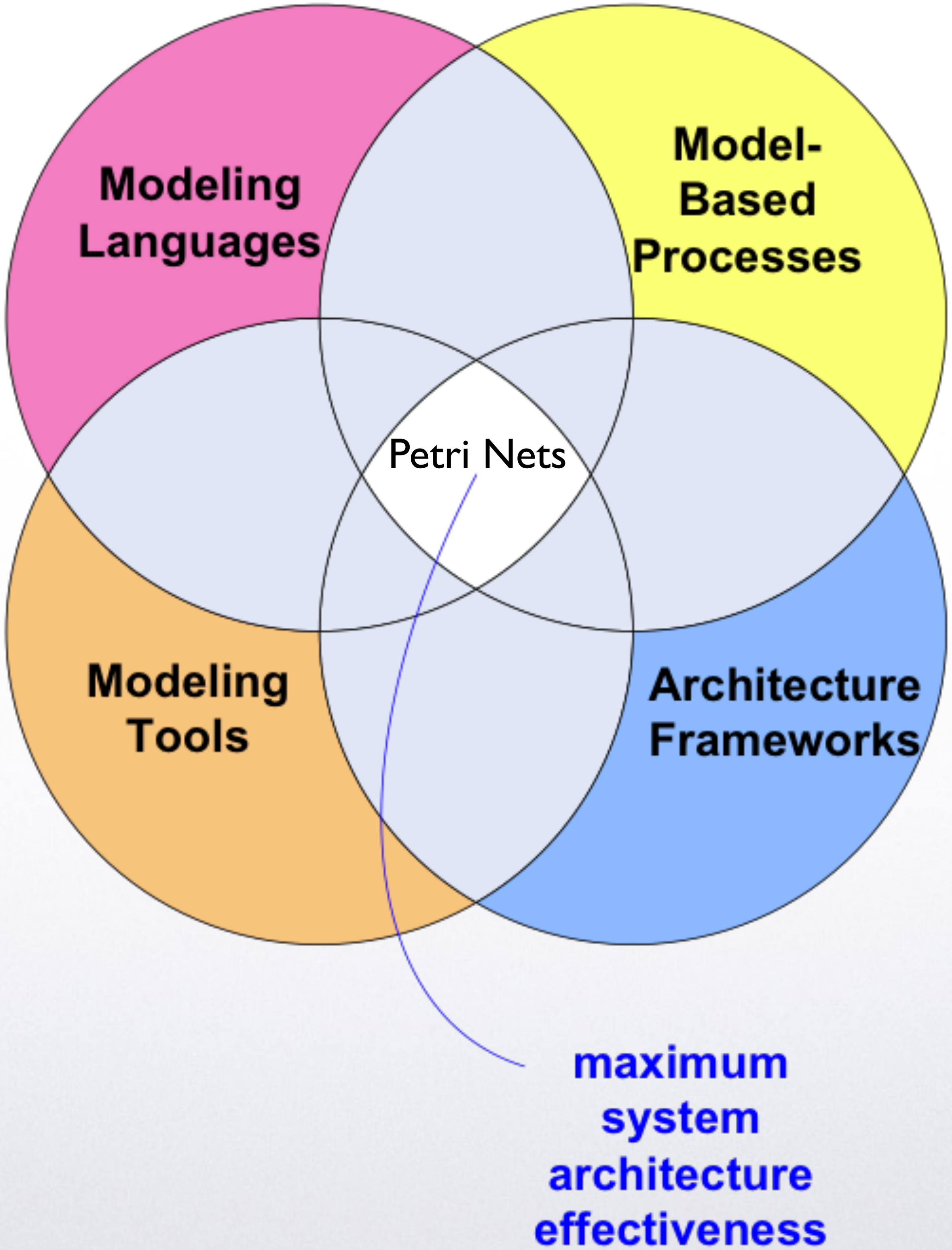

Model
Object-oriented
Event-oriented
Hybrid
State-oriented

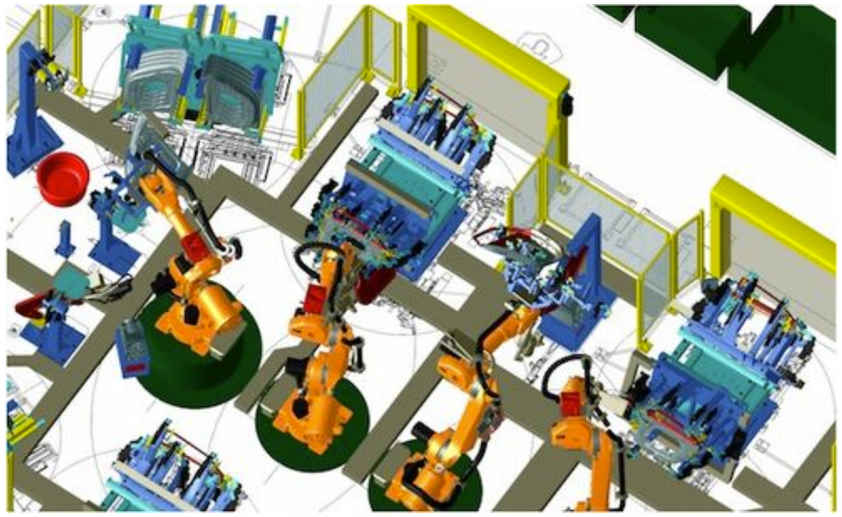
General Systems Engineering Design

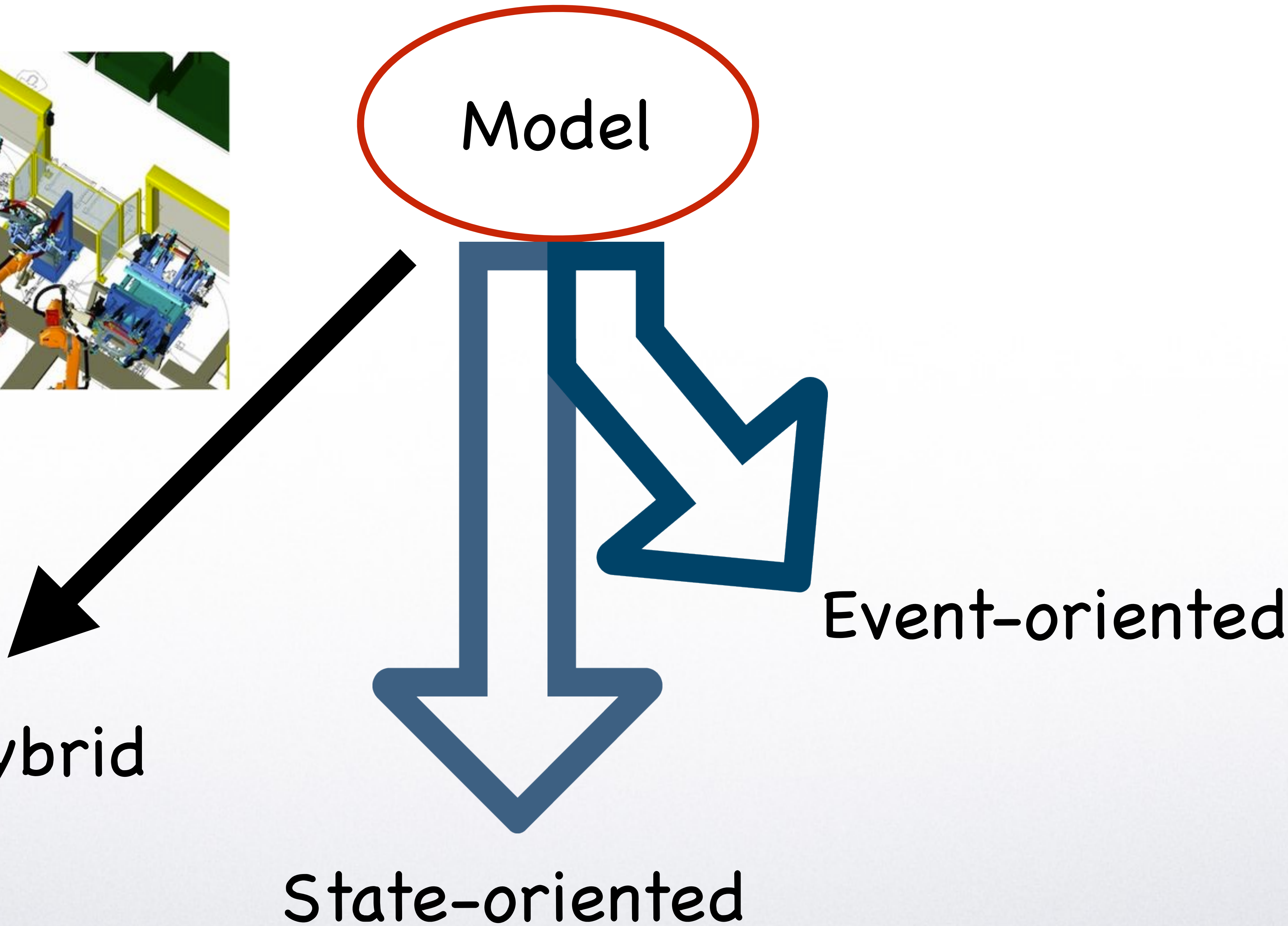
method name	paradigm	formality	graphical represen-tation	object-oriented
Action Systems	state transition	formal	no	no
B	state transition	formal	no	no
CASL	algebra	formal	no	yes
Cleanroom & JSD	traces & process algebra	formal	yes	no
COQ	state transition	formal	no	no
Estelle	state transition	formal	no	no
LOTOS	process algebra	formal	no	yes
OMT & B	state transition	formal	yes	yes
Petri Nets	state transition	formal	yes	no
Petri Nets with Objects	state transition	formal	yes	yes
SART	state transition	informal & semi-formal	yes	no
SAZ	state transition	semi-formal & formal	yes	no
SCCS	process alge-bra	formal	no	no
SDL	state transition	formal	yes	yes
UML	state transition	informal & semi-formal	yes	yes
VHDL	state transition	formal	no	no
Z	state transition	formal	no	no

method name	paradigm	formality	graphical represen-tation	object-oriented
B	state transition	formal	no	no
OMT & B	state transition	formal	yes	yes
Petri Nets	state transition	formal	yes	no
Petri Nets with Objects	state transition	formal	yes	yes

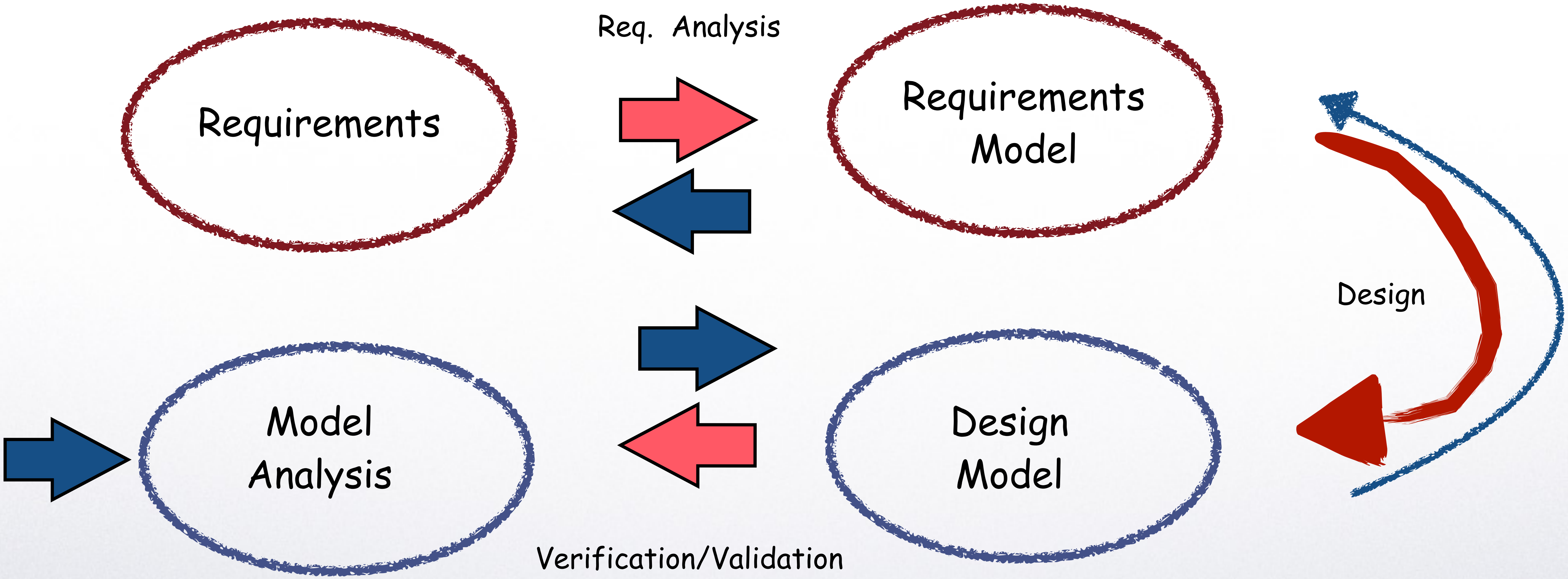
method name	logic	provability	model checking	event inhi-bition
B	yes	yes	yes	no
OMT & B	yes	yes	yes	no
Petri Nets	no	yes	yes	no
Petri Nets with Objects	no	yes	yes	no


Modeling Languages
Model-Based Processes
Petri Nets
Modeling Tools
Architecture Frameworks
maximum system architecture effectiveness



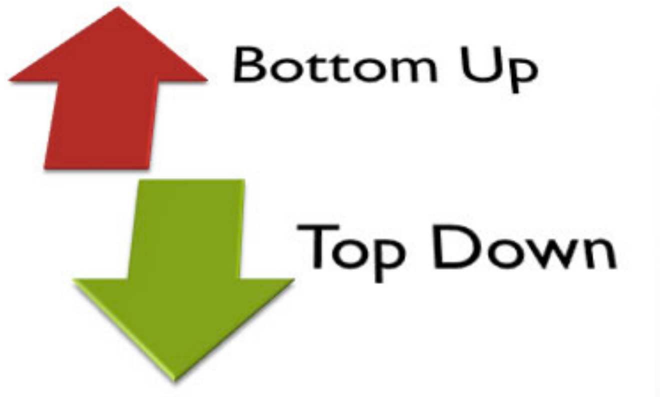

Model
Event-oriented
Hybrid
State-oriented

State Space PN modeling is problem-oriented.



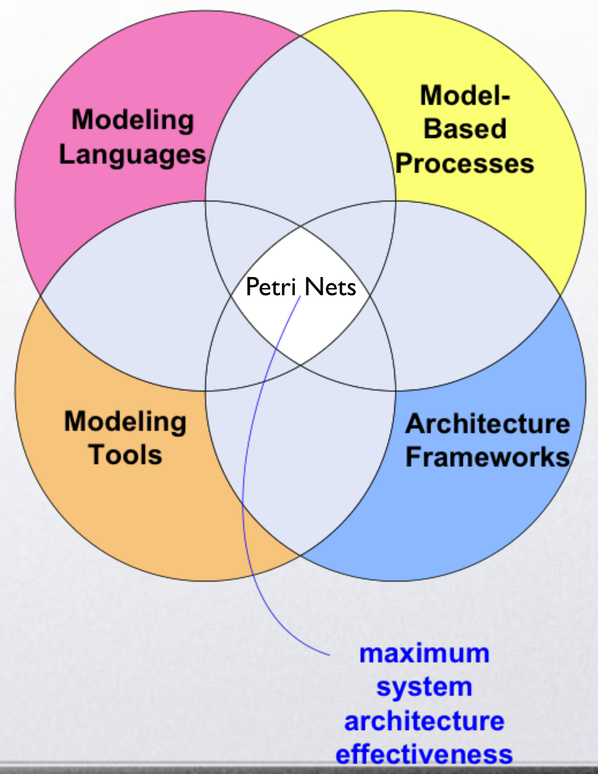
Systems Design

Structured and Systemic Method



Problem Solving Method

Formal Methods



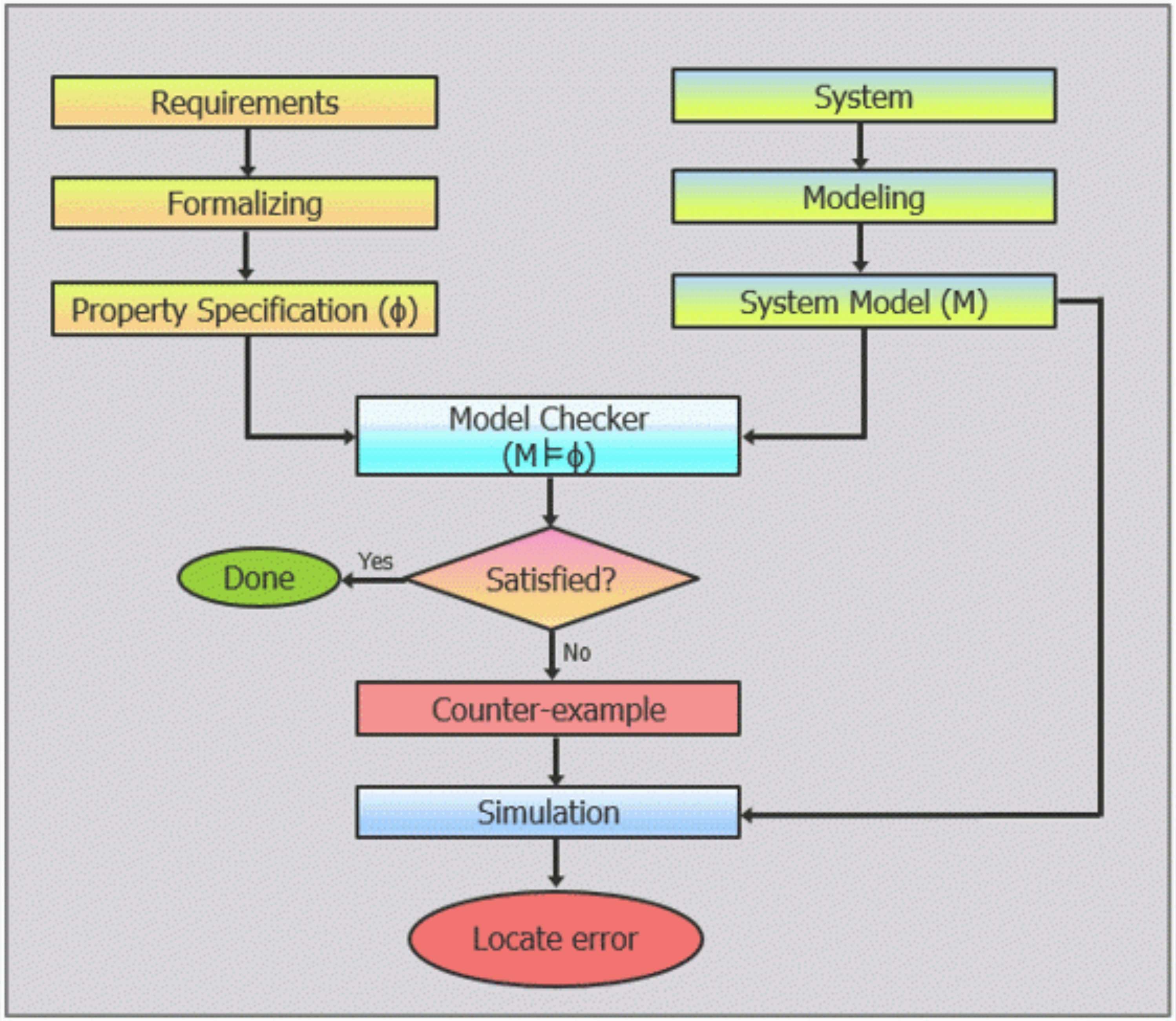
Some assumptions: fairness and determinism

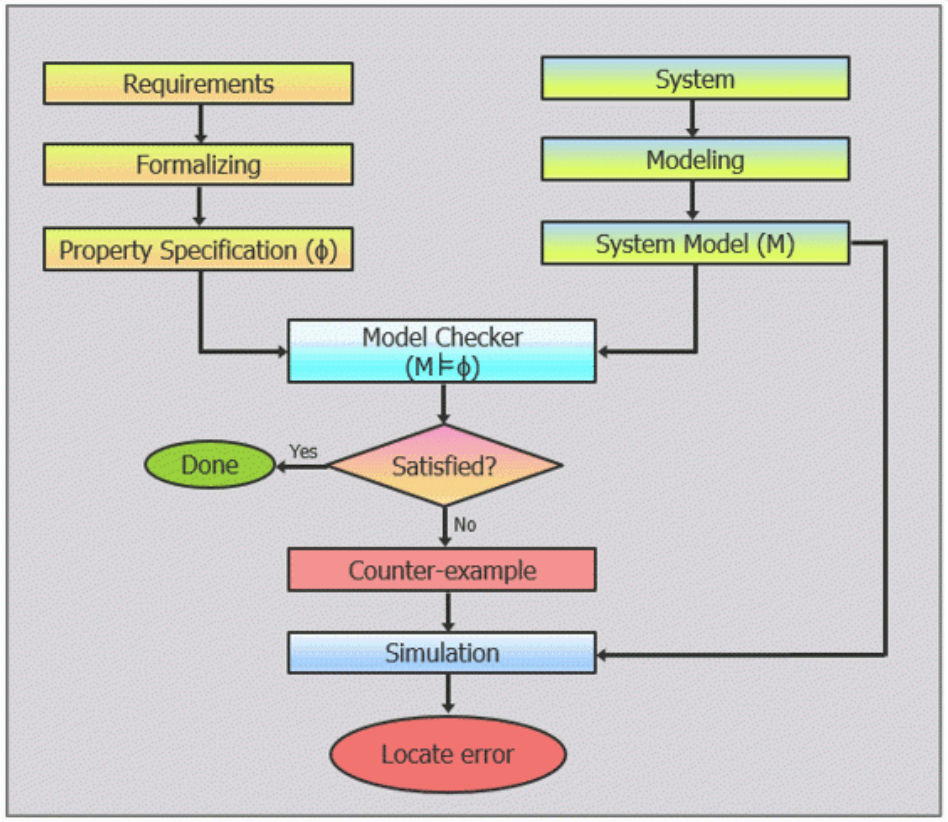
Weak-fair transitions

Productive transitions eventually occur in an infinite occurrence sequence if they are persistently enabled. This characteristic is also known as the finite delay property or as the weak fair condition. In an infinite sequence, productive transitions must not be enabled in the last marking.

Fair transitions (also known as strongly fair) eventually occur in an infinite occurrence sequence if they are enabled infinitely often. In an infinite sequence, fair transitions must not be enabled in the last marking.

All other transitions are said normal and are not constrained.

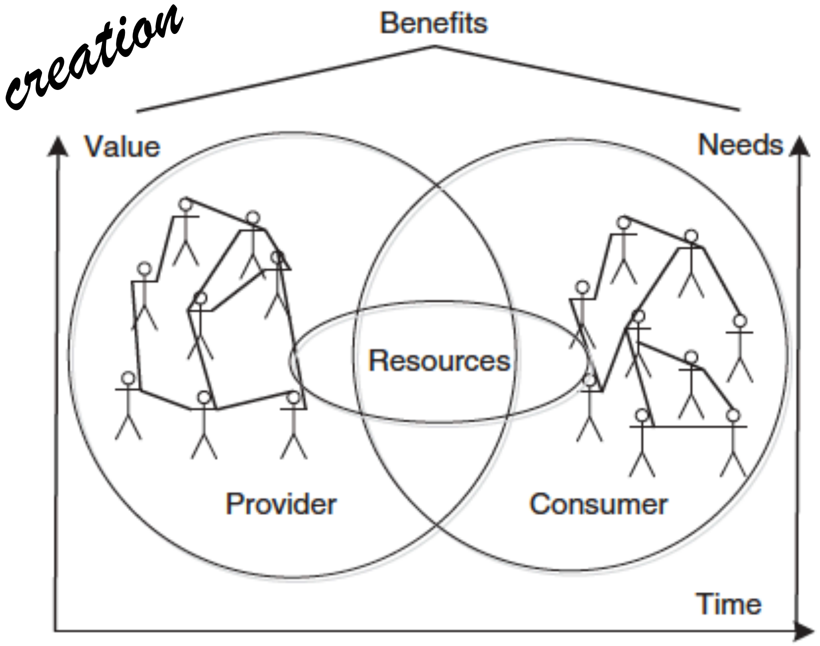

Requirements
Formalizing
Property Specification (ϕ)
System
Modeling
System Model (M)
Model Checker
(M⊨ϕ)
Satisfied?
Yes
Done
No
Counter-example
Simulation
Locate error



A formatação teórica usada comumente diz que os requisitos devem ser feitos de maneira formal, bem como o modelo do sistema. Para sistemas dinâmicos automatizados a nossa recomendação (e de vários pesquisadores no mundo) é que ambas sejam feitas em redes de Petri.

The need for requirements formal modeling and verification

value co-creation



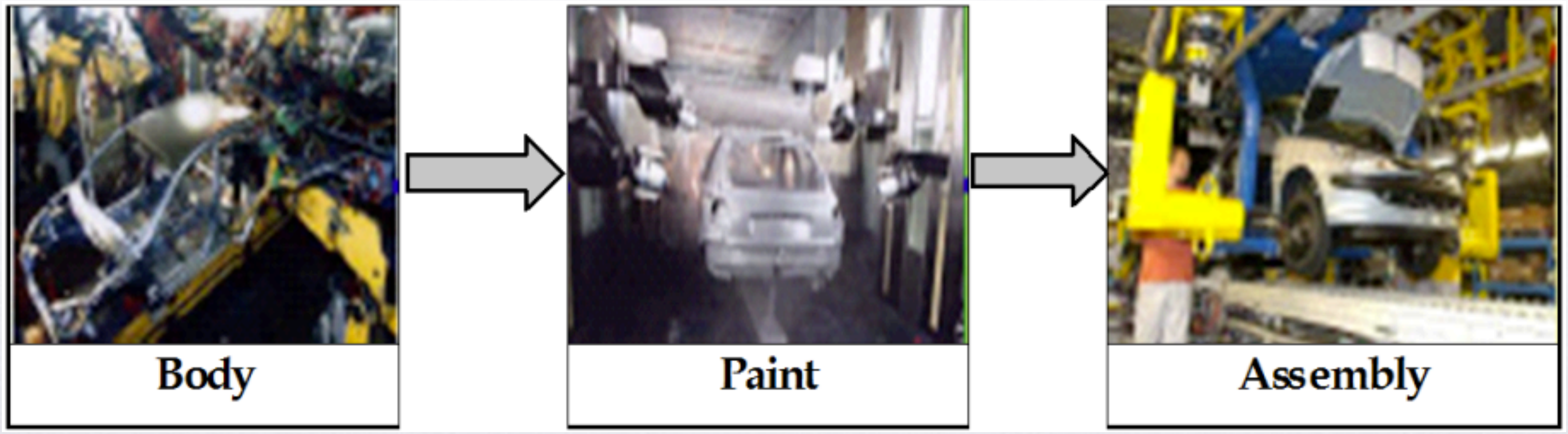
Robin Qiu, Service Science, Wiley, 2014

duality law $\neg \bigcirc \varphi \equiv \bigcirc \neg \varphi$ $\neg \Diamond \varphi \equiv \Box \neg \varphi$ $\neg \Box \varphi \equiv \Diamond \neg \varphi$	*idempotency law* $\Diamond \Diamond \varphi \equiv \Diamond \varphi$ $\Box \Box \varphi \equiv \Box \varphi$ $\varphi \, \mathsf{U} \, (\varphi \, \mathsf{U} \, \psi) \equiv \varphi \, \mathsf{U} \, \psi$ $(\varphi \, \mathsf{U} \, \psi) \, \mathsf{U} \, \psi \equiv \varphi \, \mathsf{U} \, \psi$
absorption law $\Diamond \Box \Diamond \varphi \equiv \Box \Diamond \varphi$ $\Box \Diamond \Box \varphi \equiv \Diamond \Box \varphi$	*expansion law* $\varphi \, \mathsf{U} \, \psi \equiv \psi \vee (\varphi \wedge \bigcirc (\varphi \, \mathsf{U} \, \psi))$ $\Diamond \psi \equiv \psi \vee \bigcirc \Diamond \psi$ $\Box \psi \equiv \psi \wedge \bigcirc \Box \psi$
distributive law $\bigcirc (\varphi \, \mathsf{U} \, \psi) \equiv (\bigcirc \varphi) \, \mathsf{U} \, (\bigcirc \psi)$ $\Diamond (\varphi \vee \psi) \equiv \Diamond \varphi \vee \Diamond \psi$ $\Box (\varphi \wedge \psi) \equiv \Box \varphi \wedge \Box \psi$	

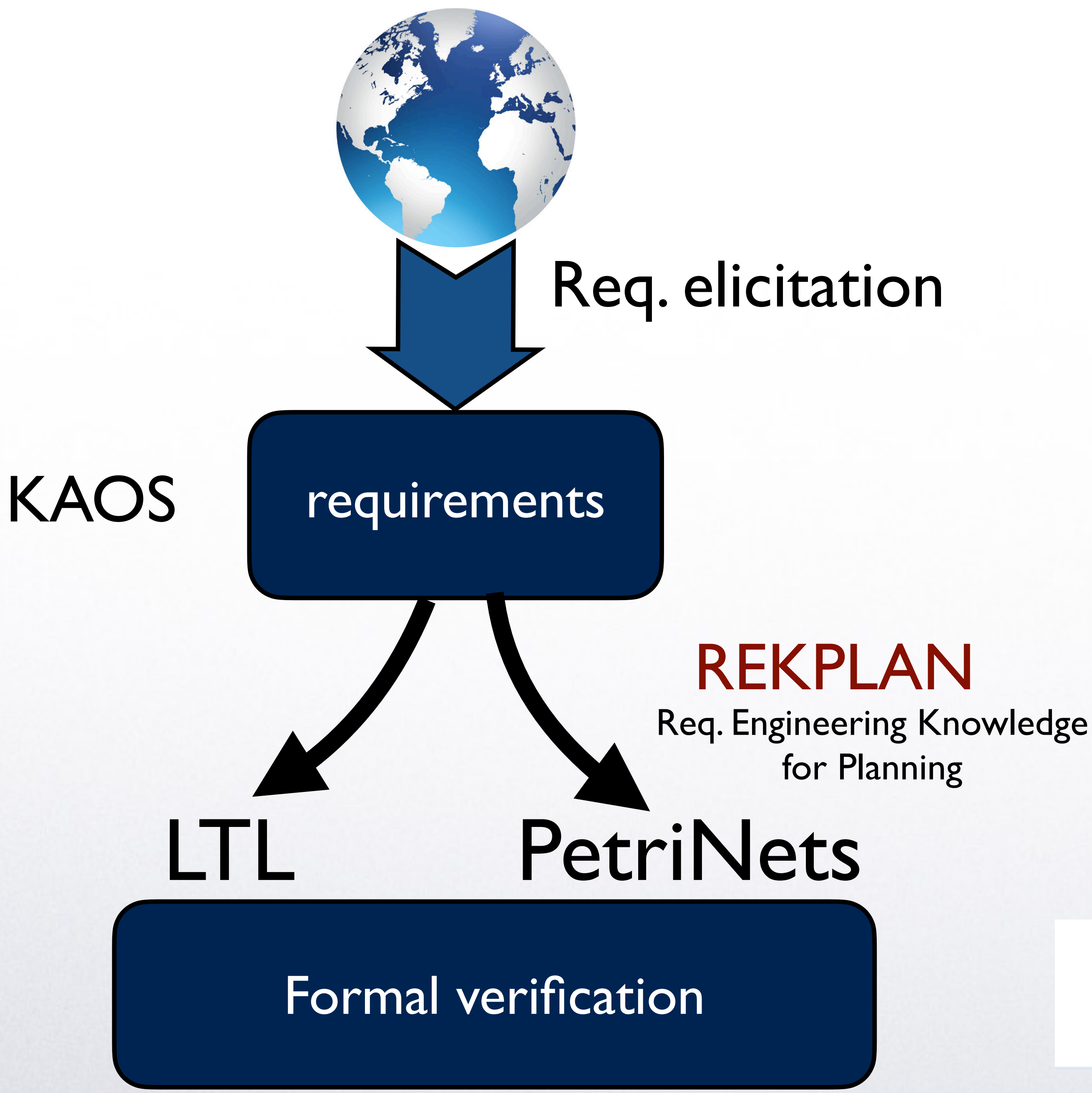
In the new design scenario, requirements should be modeled (following an MBE approach) and verified before going further with the design process.

A manufacturing example

ROADEF is a challenge repositories were real companies leave some challenges to be solved to academics. Car sequencing was proposed in 2005 by RENAULT and were used in several competitions of knowledge engineering systems later on.

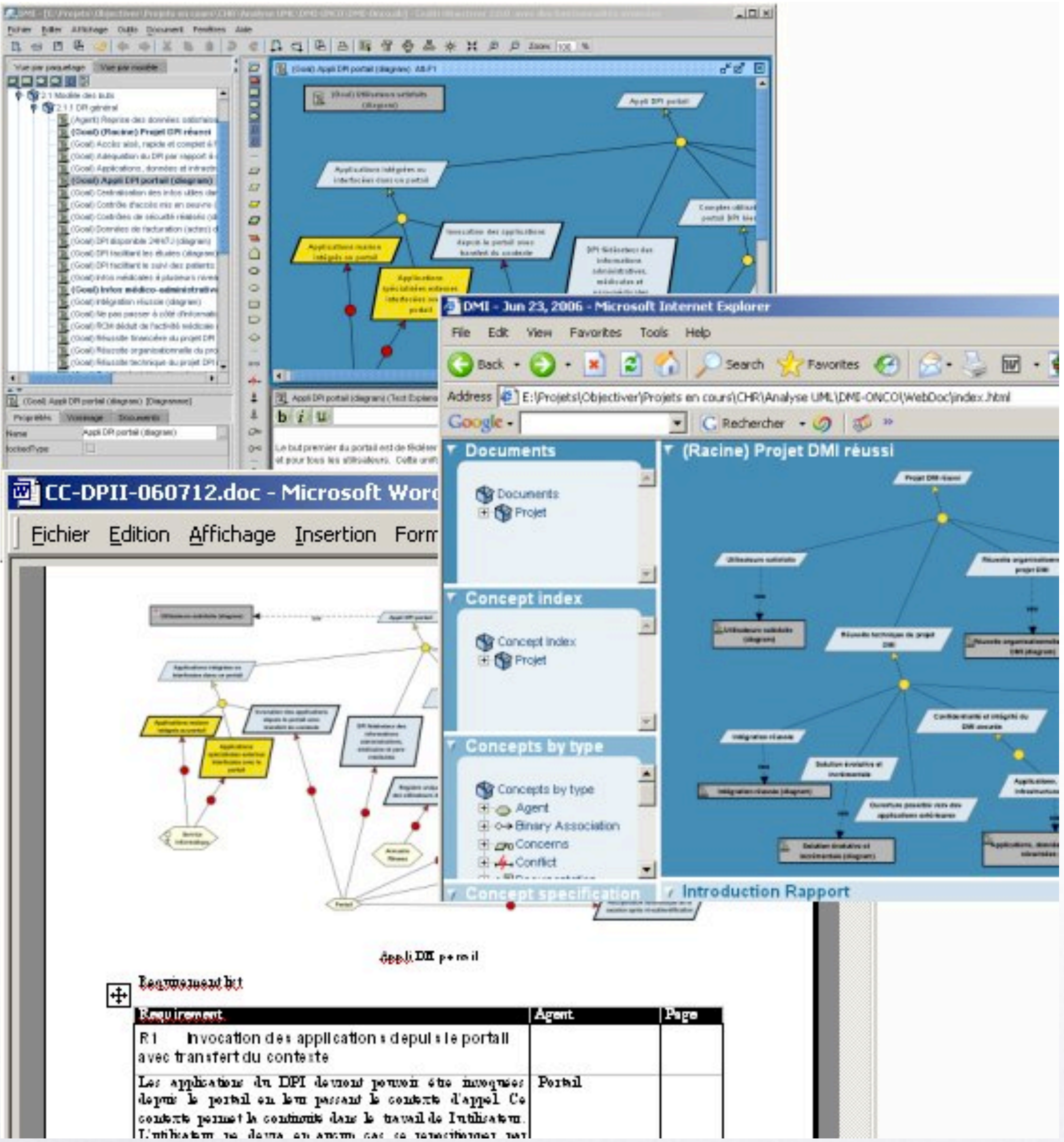


The proposed method

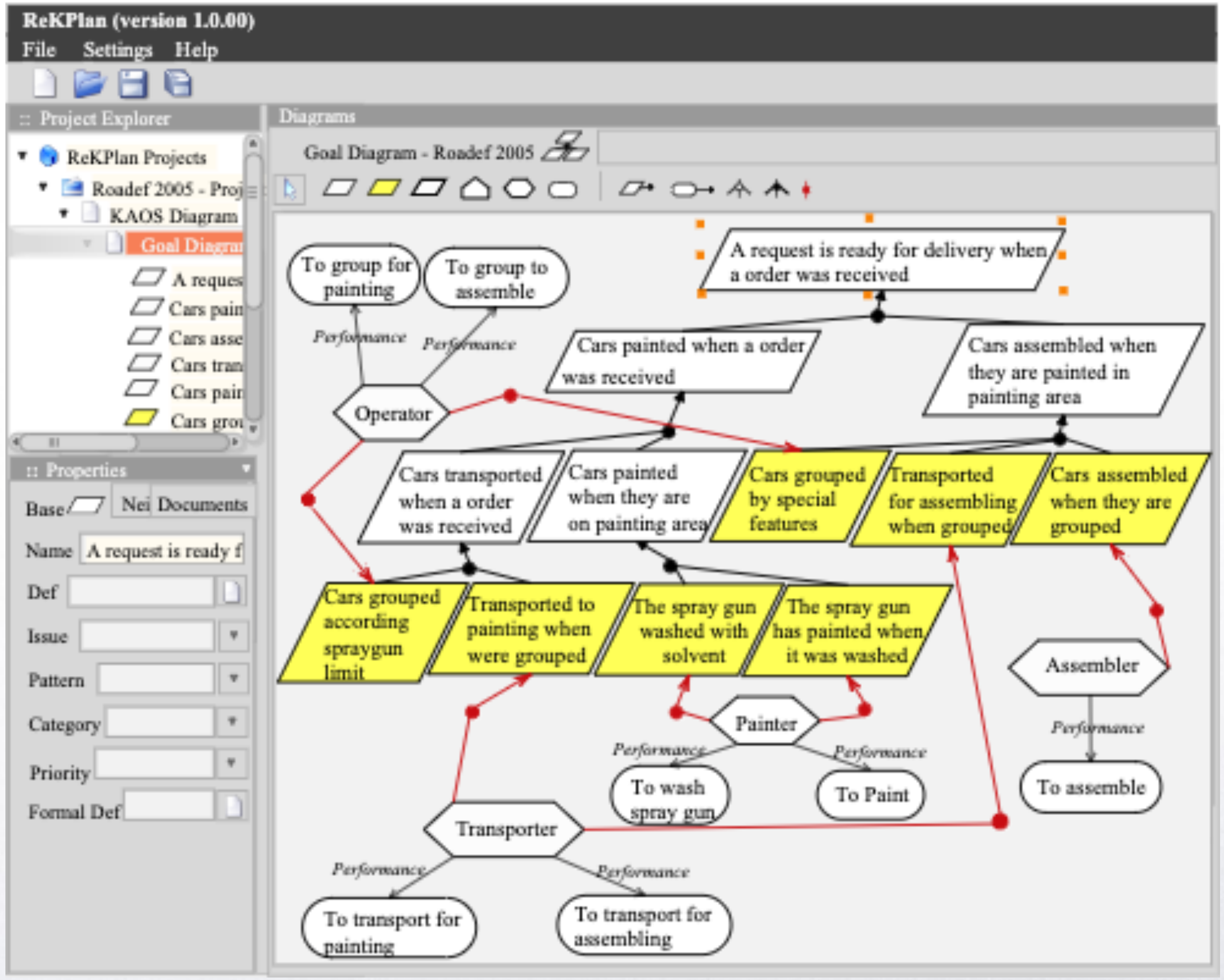


Documentation

Respect-IT
Requirements Engineering &
SPECification Techniques
for InformationTechnology



Objectiver



REKPLAN

Converting KAOS to Petri nets

Element	Name
	Box
	Activity
	Pseudo-Box
	Arc
	Enabled Arc
	Macro-Box
	Macro-Activity

Fig. 2. GHENeSys is a unified net that incorporate - formally - extensions to Place/Transition and High-Level Net definitions as Pseudo-box, to represent observable but not controllable events and hierarchical elements (macro-box and macro-activities).

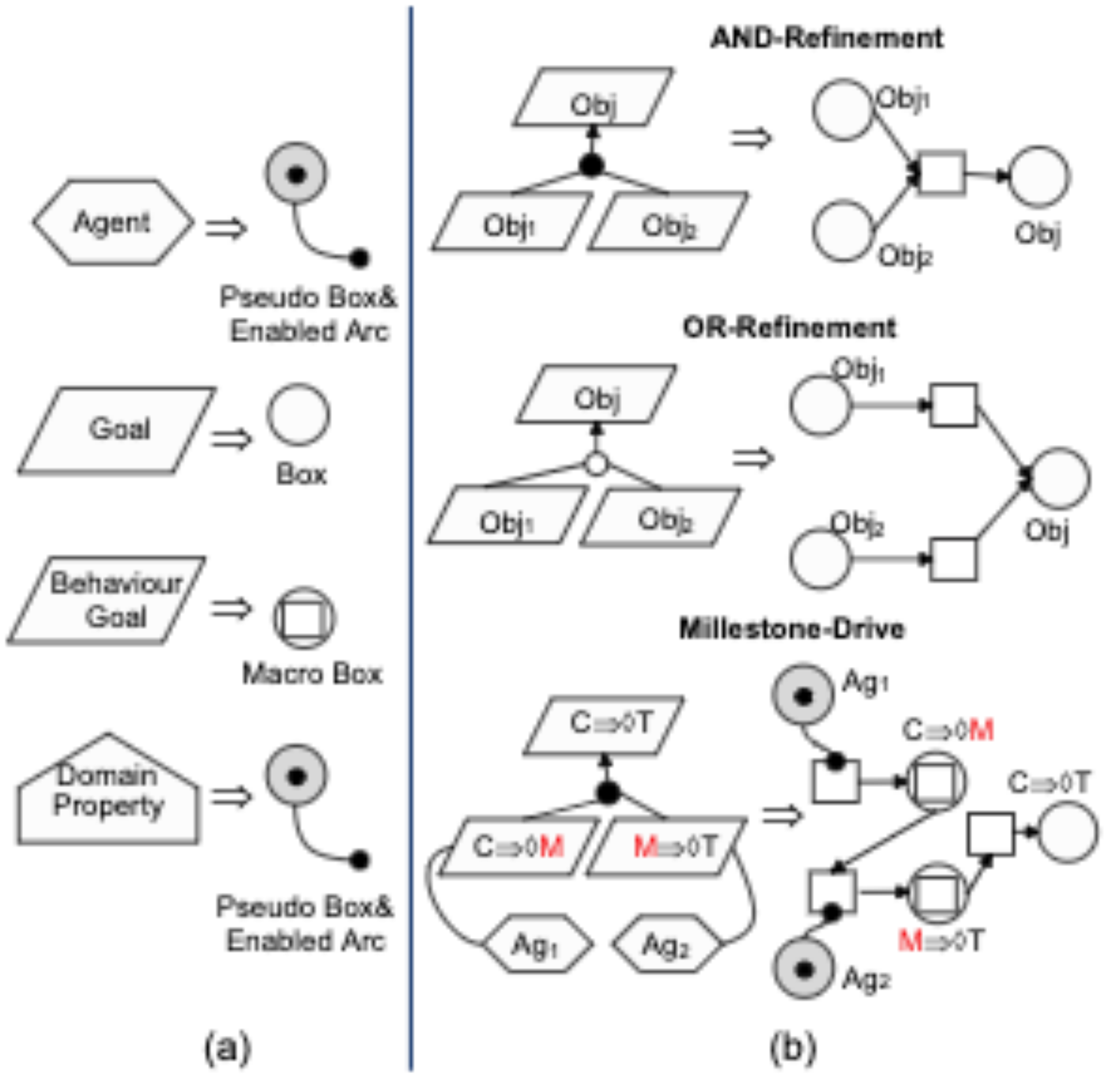


Fig. 6. a) Semantic Translation: basic elements. b) Semantic translation for refinements.

New approach to
manufacturing & assembling design should be:

- Systemic;
- Free from functional/non-functional dilema;
- Device oriented;
- Attached to service;
- Distributed;
- Flexible.

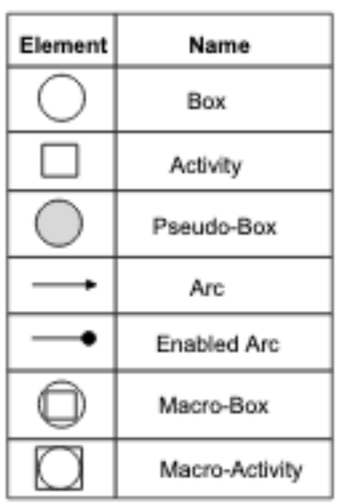
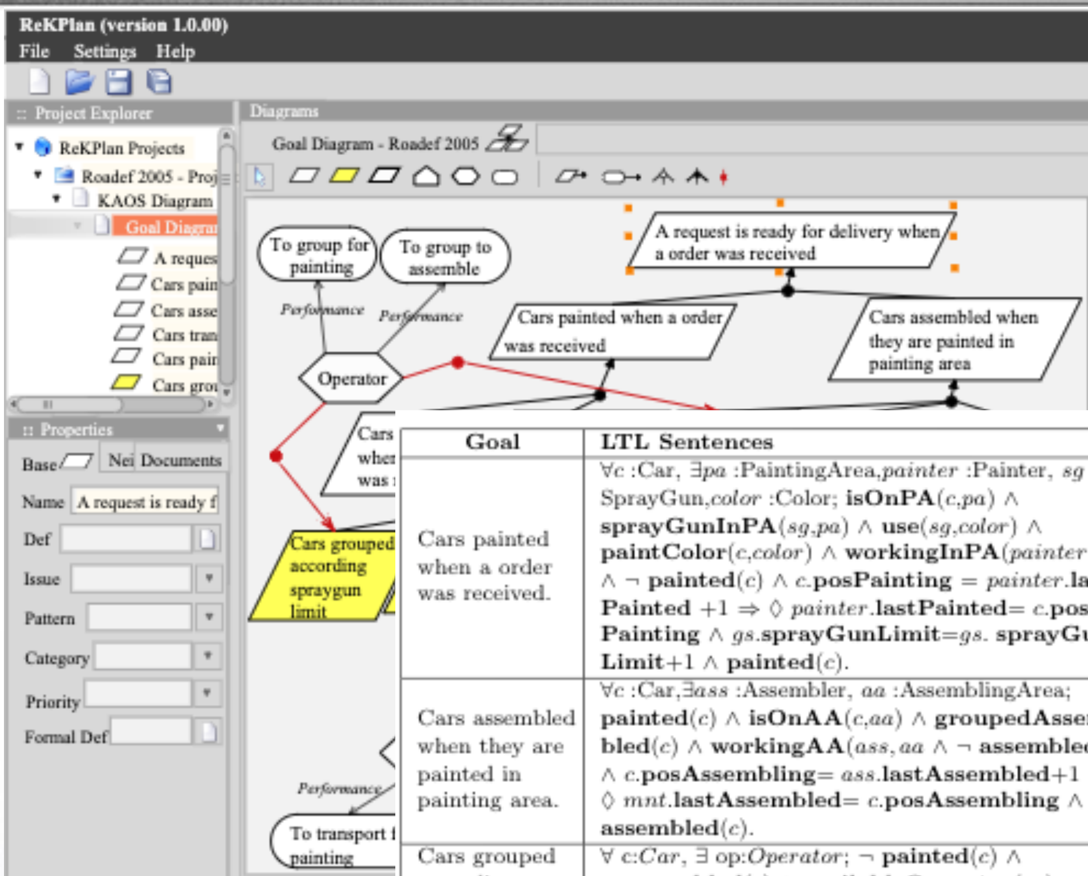


Fig. 2. GHENeSys is a unified net that incorporate - formally - extensions to Place/Transition and High-Level Net definitions as Pseudo-box, to represent observable but not controllable events and hierarchical elements (macro-box and macro-activities).

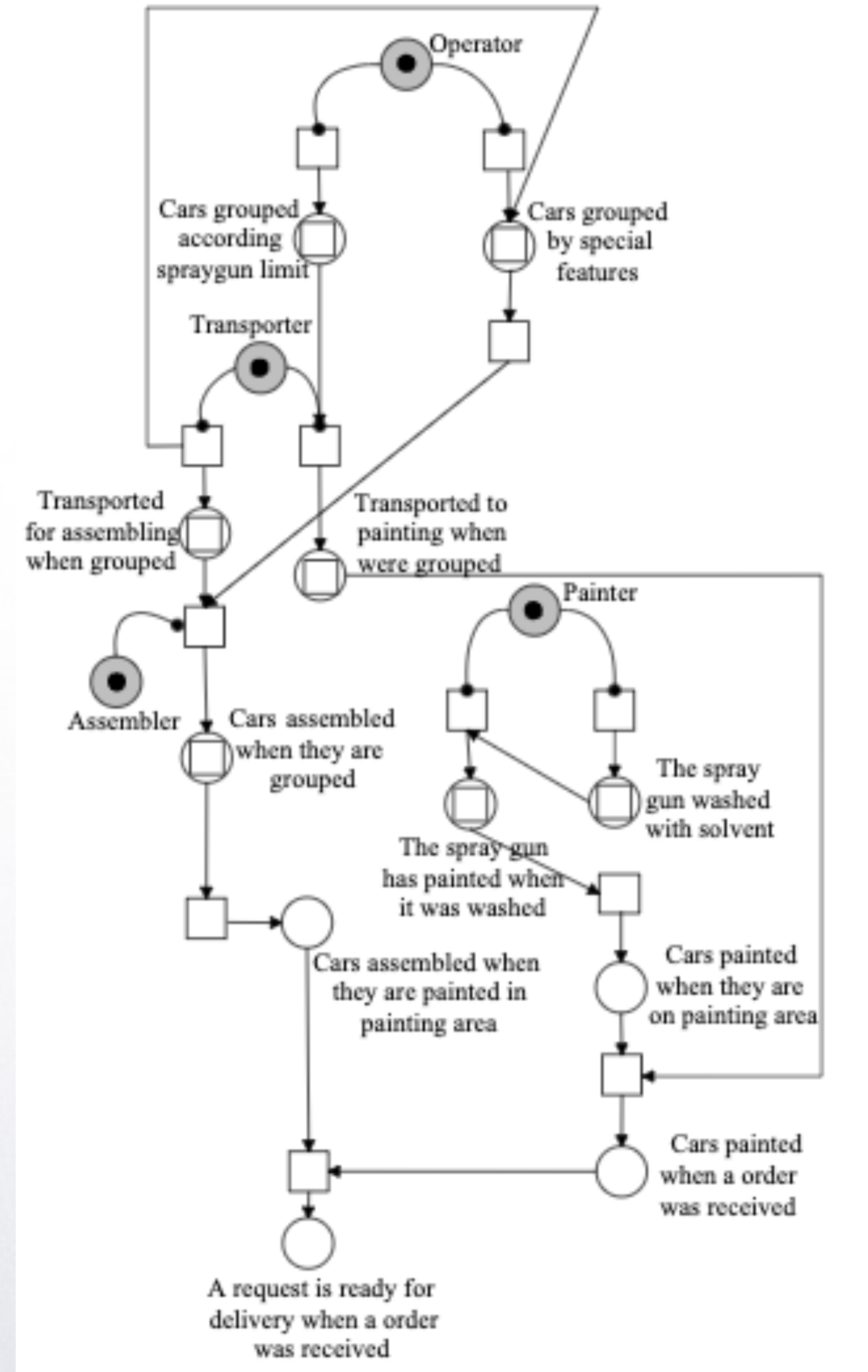
Petri Nets (PNs)

- Model introduced by C.A. Petri in 1962
  - Ph.D. Thesis: "Communication with Automata"
- Applications: distributed computing, manufacturing, control, communication networks, transportation…
- PNs describe explicitly and graphically:
  - sequencing/causality
  - conflict/non-deterministic choice
  - concurrency
- Asynchronous model
- Main drawback: ~~no hierarchy~~

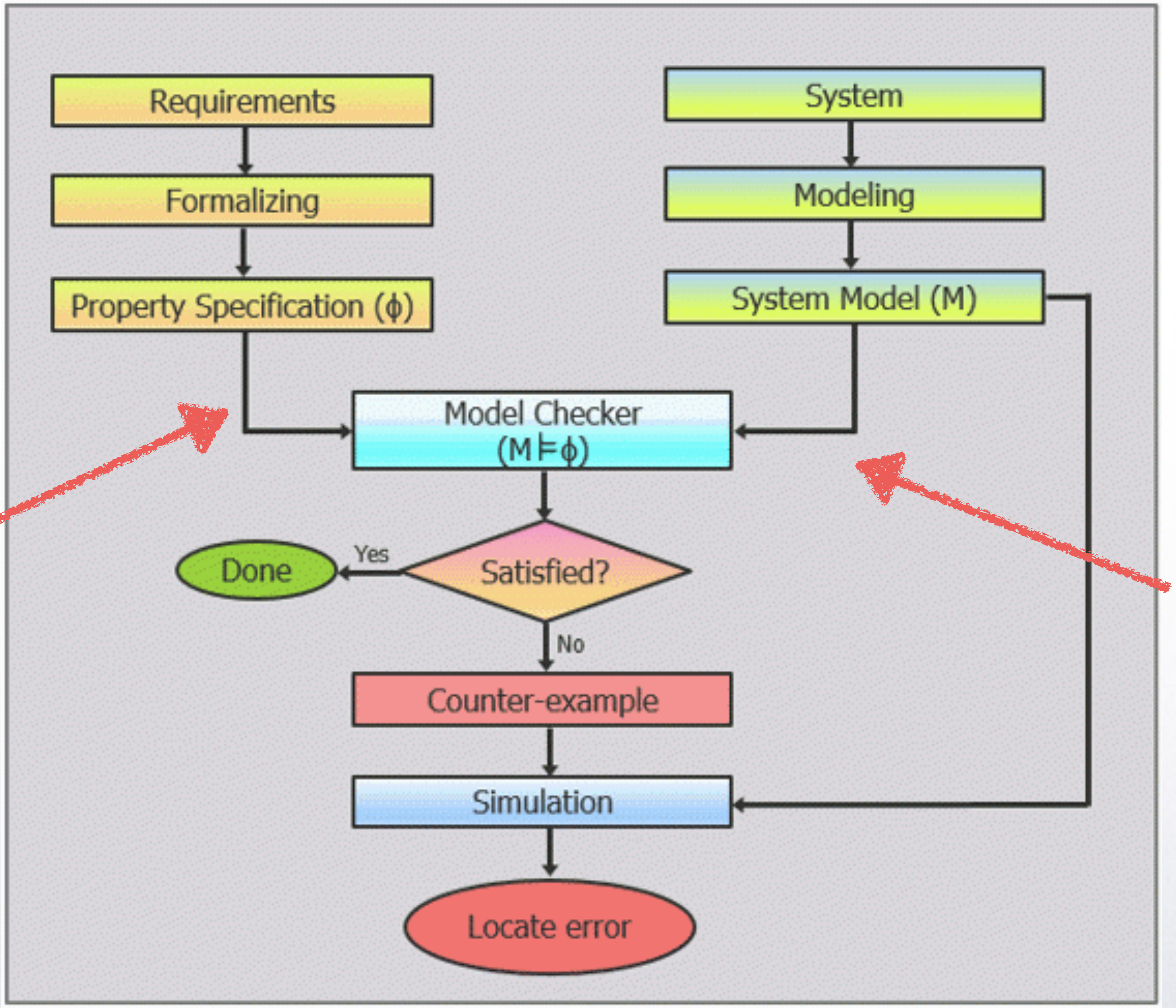
4



Goal	LTL Sentences
Cars painted when a order was received.	$\forall c$:Car, $\exists pa$:PaintingArea,$painter$:Painter, sg : SprayGun,$color$: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.	$\forall c$:Car,$\exists ass$:Assembler, aa :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, $\exists$ 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.	$\forall c$:Car, $\exists$ 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.	$\forall c$:Car, $\exists tra$:Transporterr, $\exists pa$:PaintingArea, sg :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:Car, $\exists tra$:Transporter, aa :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 :SprayGun, $\exists$ 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, sg:$SprayGun$, $\exists$ painter:$Painter$; $\mathbf{has}(painter, sg) \wedge \mathbf{clean}(sg) \Rightarrow \Diamond\, \neg\, \mathbf{clean}(sg)$.



Petri nets

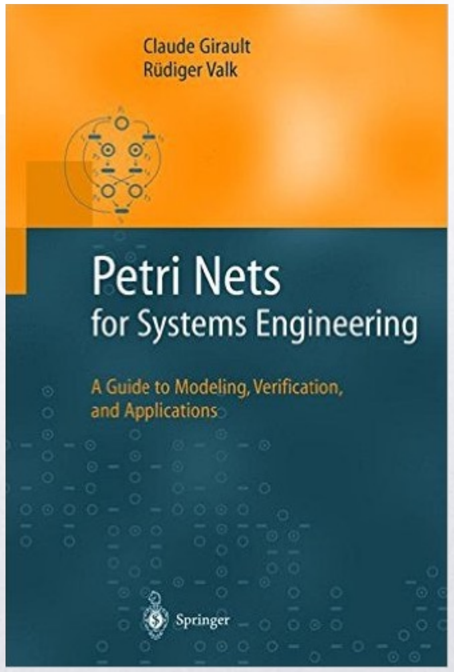


Requirements
Formalizing
Property Specification (ϕ)
System
Modeling
System Model (M)
Model Checker
(M⊨ϕ)
Done
Yes
Satisfied?
No
Counter-example
Simulation
Locate error

?

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.



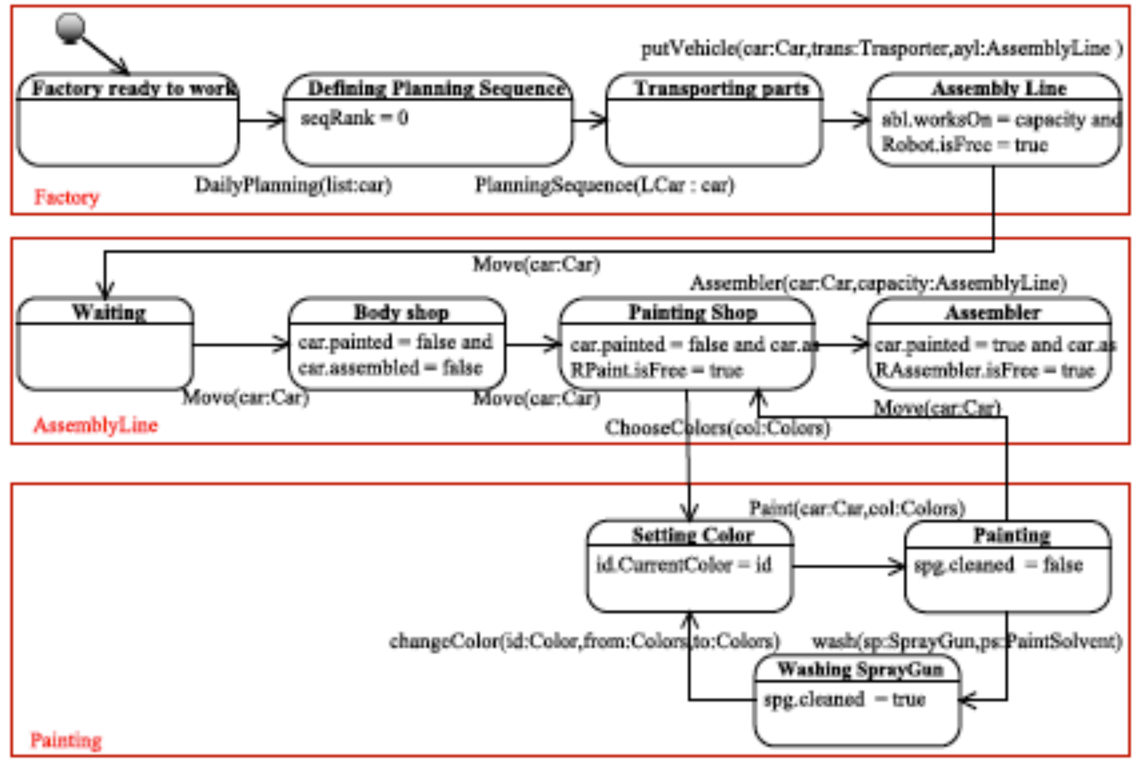


Fig. 9. Behavioral State diagram for car sequence problem.

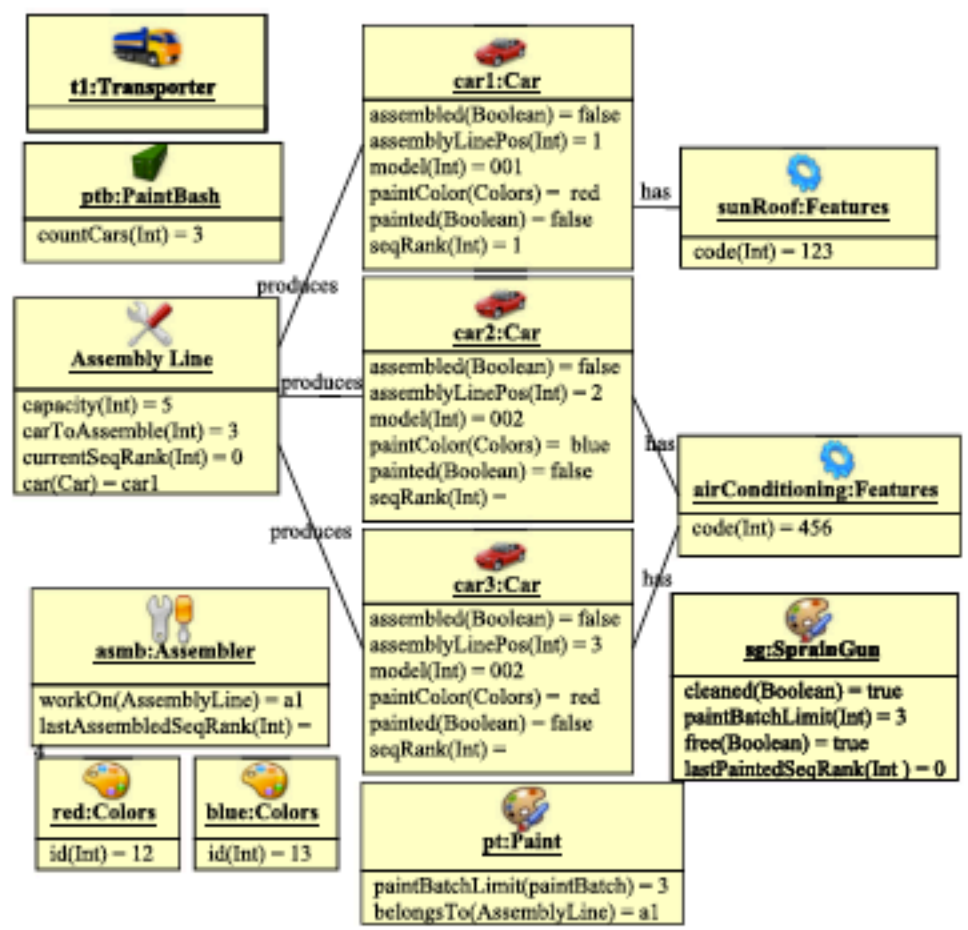


Fig. 10. Object diagram for initial state — ROADEF challenge 2005.

Engineering Applications of Artificial Intelligence 81 (2019) 373–386

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

A new hierarchical approach to requirement analysis of problems in automated planning☆

Javier Martinez Silva *, José Reinaldo Silva

Department of Mechatronics Engineering, University of São Paulo, Brasil

ARTICLE INFO

Keywords:
Knowledge engineering
Hierarchical automated planning
Planning domain analysis
Hierarchical petri nets
Requirements analysis

ABSTRACT

The use of Knowledge Engineering (KE) processes to analyze and configure domains in automated planning is becoming more appealing since it was noticed that this issue could make a difference to solve real problems. The contrast between a generic domain independent approach, taken as canonical in AI, and alternative processes that include knowledge engineering – eventually adding specific knowledge – has been discussed by Computer and Engineering communities. A big impact has been noticed mainly in the early phase of requirement analysis when KE approach is normally introduced. Requirement analysis is responsible for carrying out the Knowledge modeling of both problem and work domains, which is a key issue to guide different planner algorithms to come out with efficient solutions. Also, there is the scalability issue that appear in most real problems. To face that, hierarchical methods played an important hole in the history of planning and inspired several solutions since the proposal of NONLIN in the 70's. Since then, the idea of associating hierarchical relational nets with partial ordered actions has prevailed when large systems were considered. However, there is still a gap between the hierarchical approach and the state of art of requirements analysis to allow features anticipated by KE approach to really appear in the requirements of a planning process. This paper proposes a pathway to solve this gap starting with requirements elicitation represented first in the conventional semi-formal (diagrammatic) language – UML – that is translated to Hierarchical Petri Nets (HPNs) by a new enhanced algorithm. The proposed process was installed in a software tool – developed by one of the authors – that analyzes the performance of the KE planning model: itSIMPLE (Integrated Tools Software Interface for Modeling Planning Environment). This tool was initially designed to use classic Place/Transition nets and an old version of UML (2.1). It is now enhanced to use UML 2.4 and a hierarchical Petri Net extension, also developed by the authors. Realistic examples illustrate the process which is now being applied to larger problems related to the manufacturing of car sequencing domain, one of challenge of ROADEF 2005 (French Operations Research & Decision Support Society). Finally, we consider the possibility to introduce another approach to the KE process by using KAOS (Keep All Object Satisfied) to make the planning design more accurate.

1. Introduction

Planning defines a specific type of state-transition problem where the goal is to find an admissible sequence of actions to bring the system from a given initial state to a target final state. Some approaches in the literature aim to improve the performance of intelligent automated planners by trying to optimize search algorithms for a general solution (Edelkamp and Jabbar, 2006). In addition, most existing work on AI planning use a domain independent approach where specific knowledge and restrictions of the target problem are not modeled and analyzed in the planning domain. However, even domain independent approaches lead to very smart solution frameworks – normally based on STRIPS – and in practice, can be adapted to solve real problems. In this work, a domain-independent general approach is still used as inspiration for planners algorithms, but before planners start the search for a sequence of actions that lead to the final state the whole planning domain is modeled and analyzed based on requirements (Vaquero et al., 2013b).

After extensive development combining domain independent and domain specific approaches some authors started to apply planning techniques to real world problems – as real logistic systems – with

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.engappai.2019.02.019.

* Corresponding author.
E-mail addresses: javsilva@usp.br (J.M. Silva), reinaldo@usp.br (J.R. Silva).
URLs: http://www.dlab.poli.usp.br (J.M. Silva), http://www.dlab.poli.usp.br (J.R. Silva).

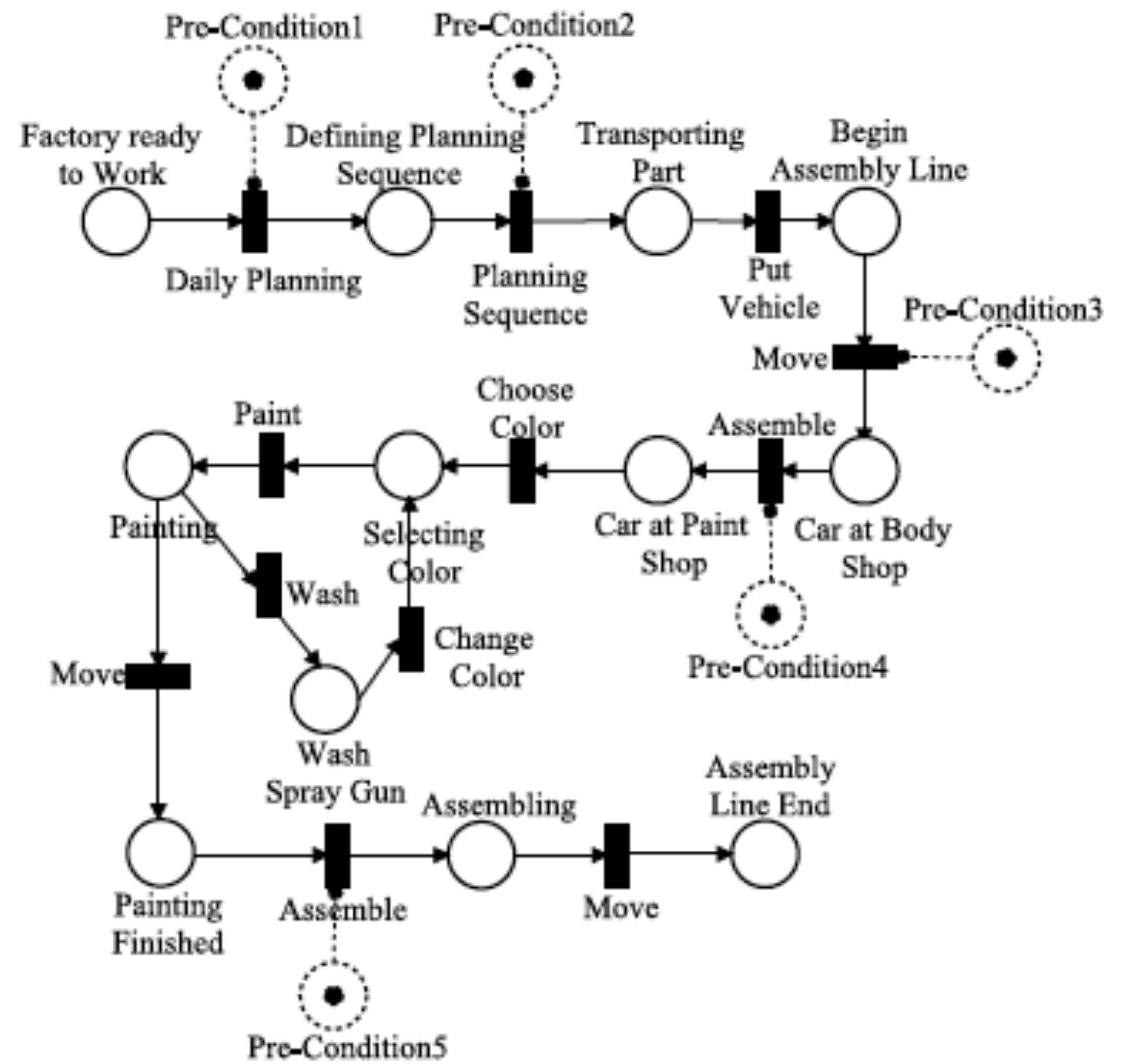
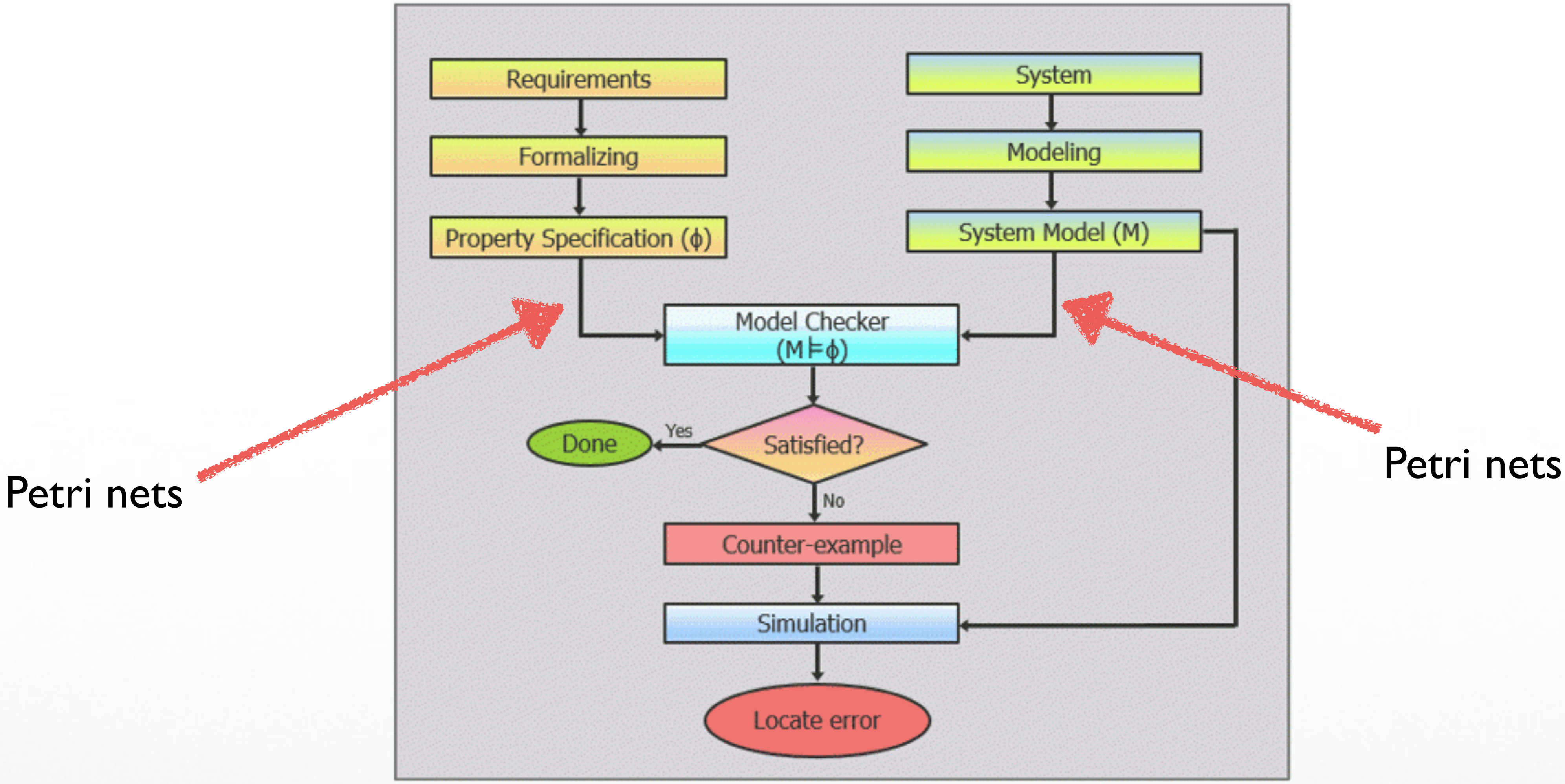


Fig. 12. Petri Net based in the Behavioral Statechart diagram for ROADEF Challenge 2005 generated by original itSIMPLE.

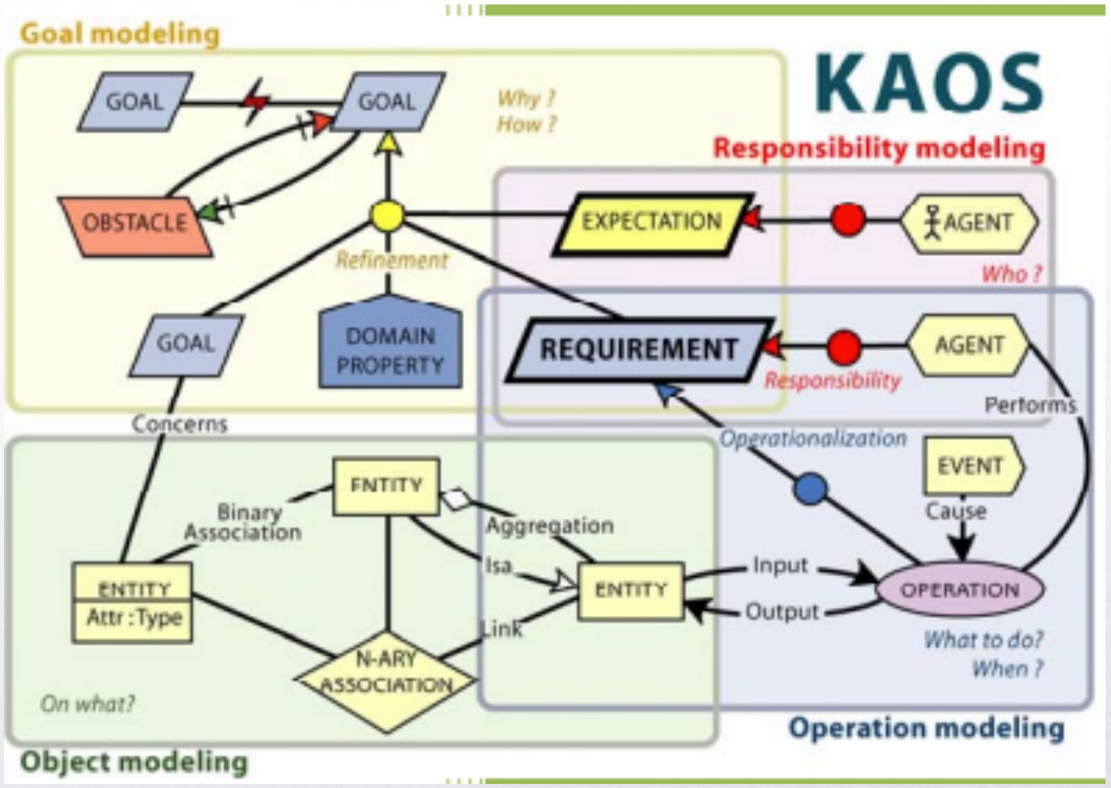
Os requisitos podem ser previamente modelados usando métodos goal-oriented ou UML, o importante é que podem ser sintetizados em redes de Petri, especialmente no caso de sistemas dinâmicos automatizados.



O matching entre o modelo de requisitos e do design da solução seria um algoritmo de construção (da solução) que respeite as restrições dos requisitos, os invariantes e que produza uma rede isomorfa à rede dos requisitos.

In the beginning of this century a new approach to Requirements Engineering was introduce: Goal-oriented Requirement Engineering (GORE) which can be a good alternative to achieve the features to service-oriented Mfg.

The new approach based on Objectives



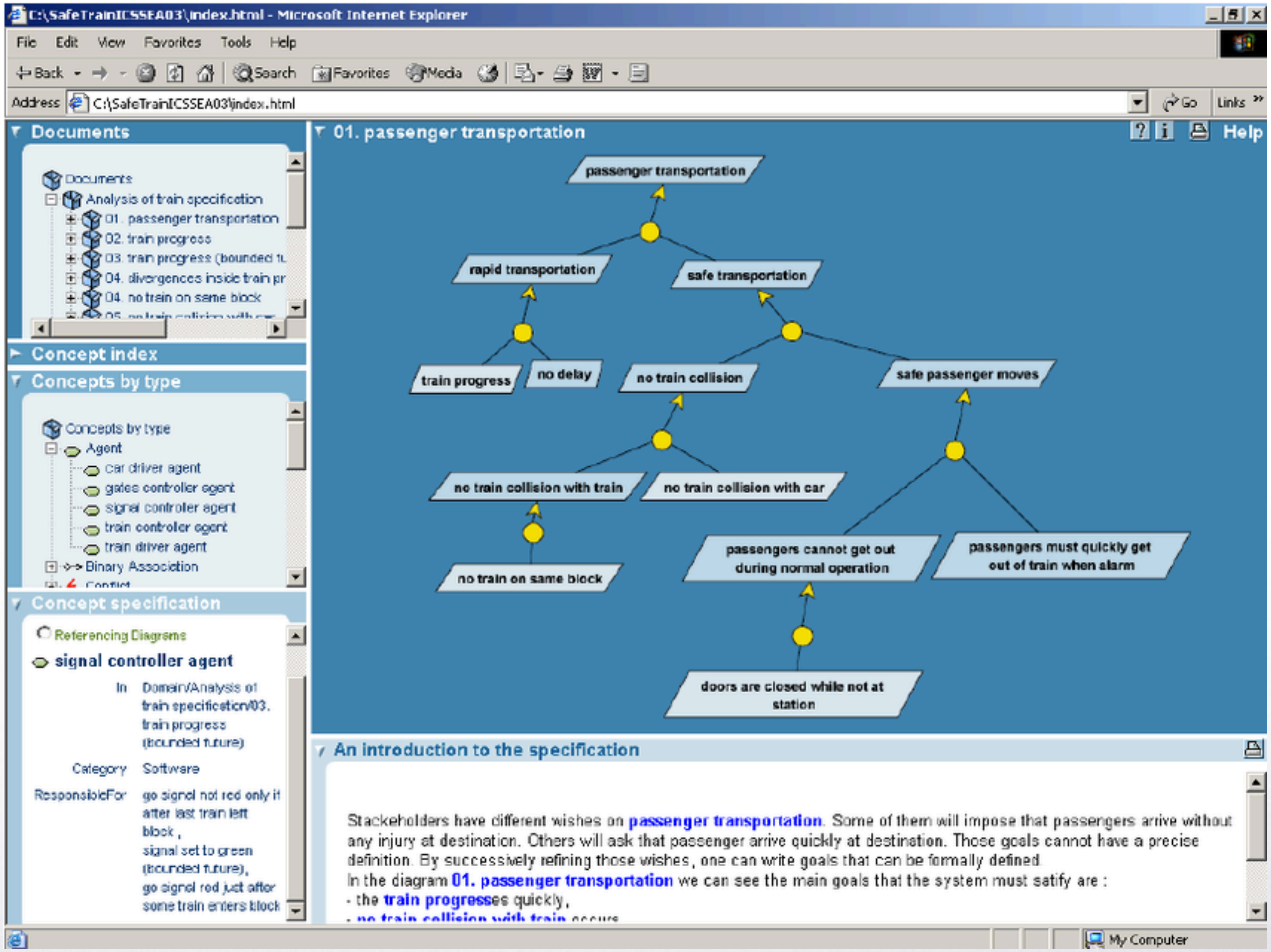
d-l@b

MECATRÔNICA

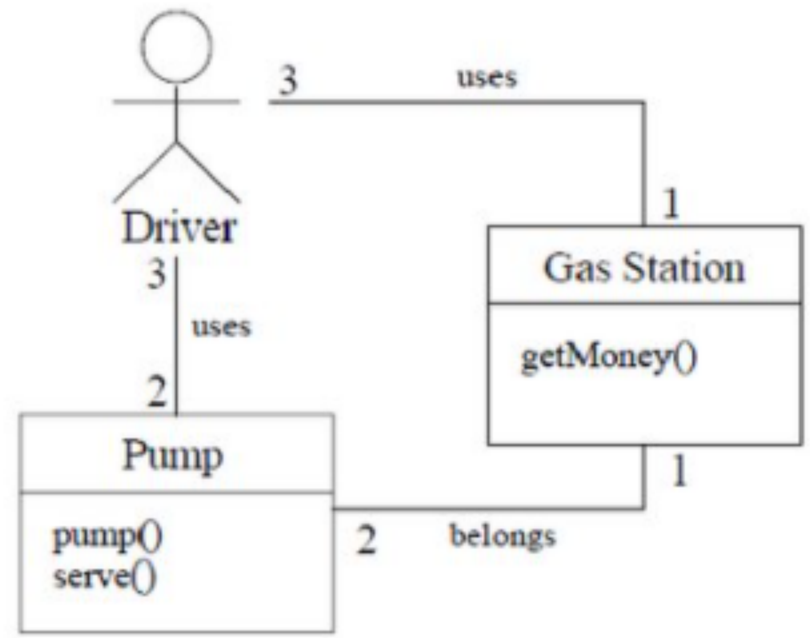
Objectiver

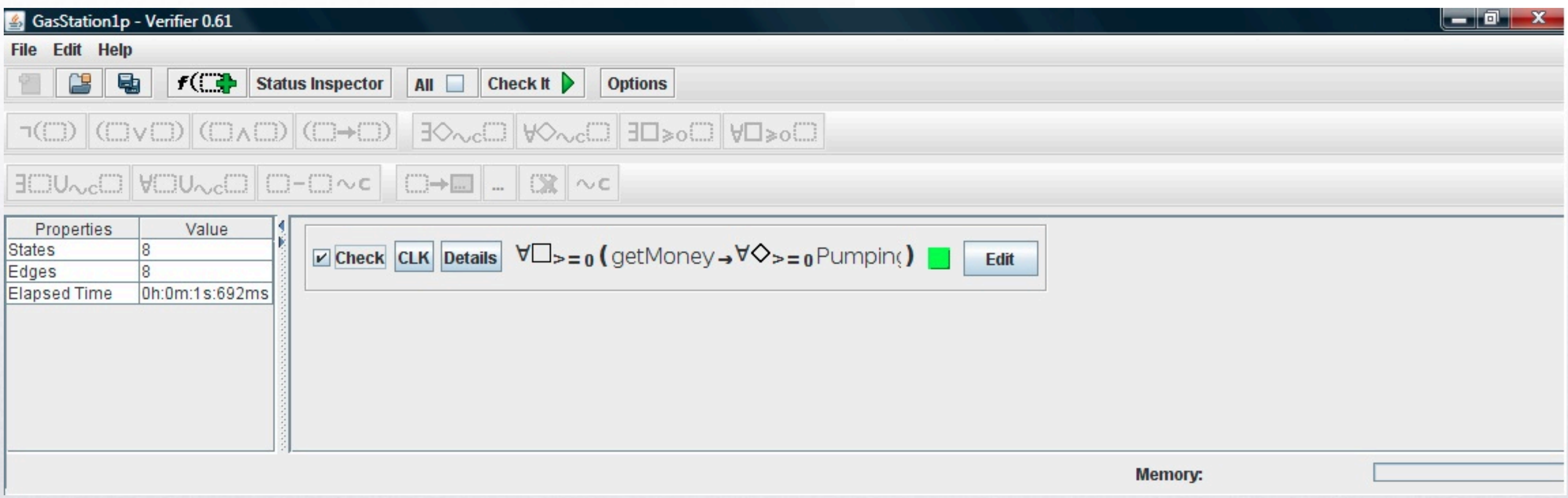
Respect-IT

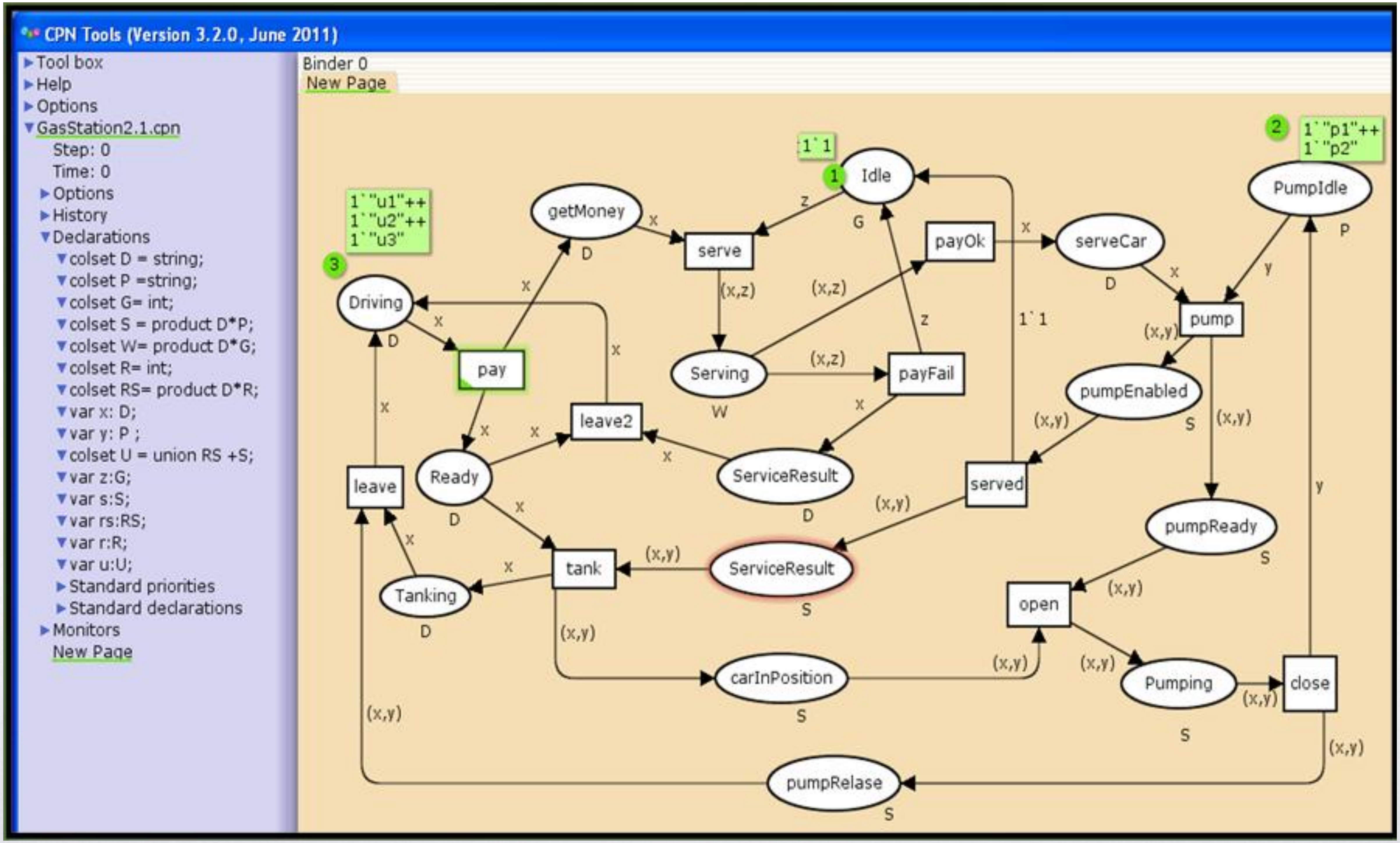
Requirements Engineering & SPECification Techniques for InformationTechnology

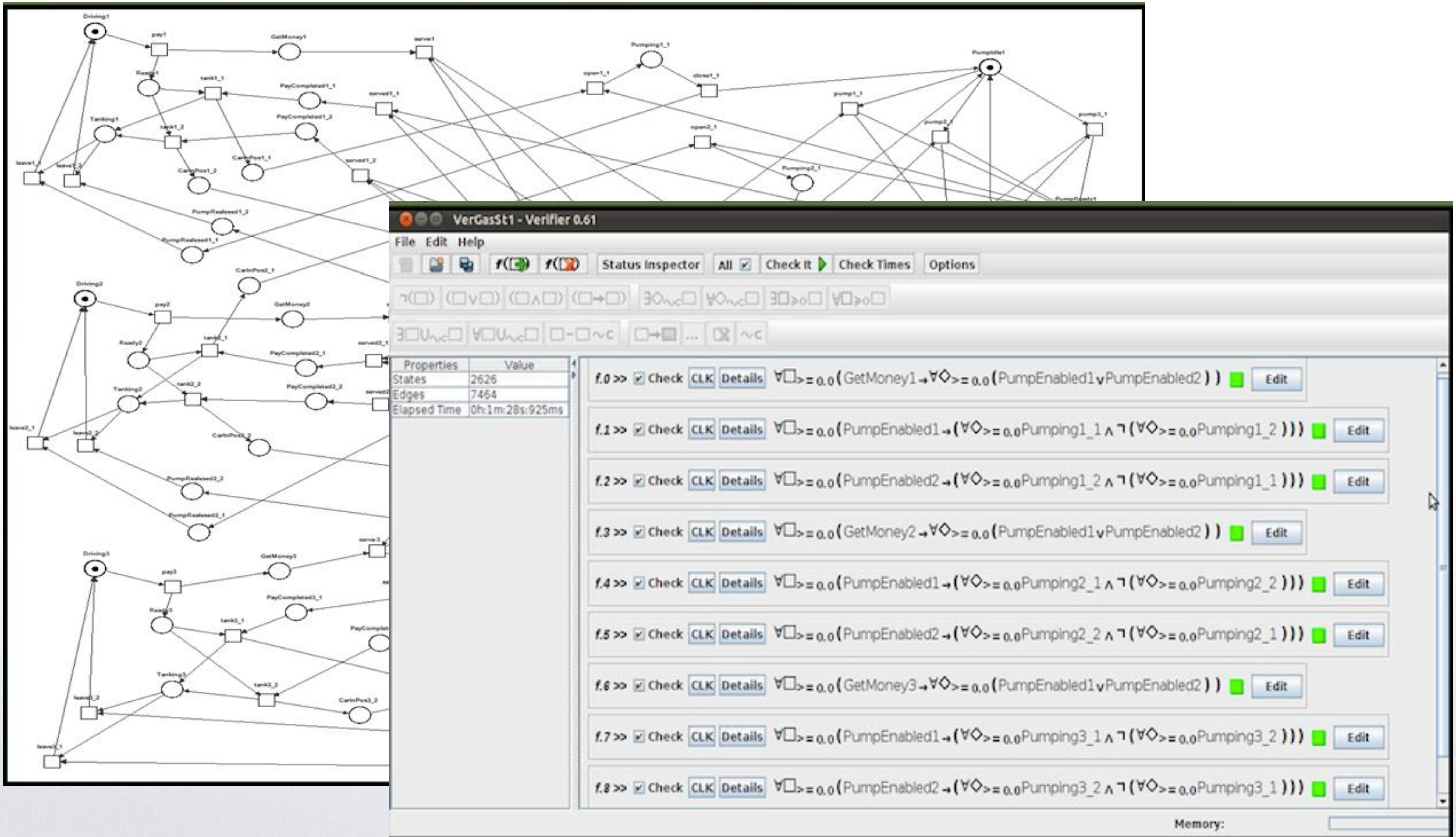


www.respect-it.com


Driver
Gas Station
getMoney()
Pump
pump()
serve()
uses
3
1
3
uses
2
1
2
belongs


GasStation1p - Verifier 0.61
File Edit Help
Status Inspector
All
Check It
Options
Properties Value
States 8
Edges 8
Elapsed Time 0h:0m:1s:692ms
Check CLK Details
∀□>=0 (getMoney → ∀◇>=0 Pumpin( )
Edit
Memory:


CPN Tools (Version 3.2.0, June 2011)
Tool box
Help
Options
GasStation2.1.cpn
Step: 0
Time: 0
Options
History
Declarations
colset D = string;
colset P =string;
colset G= int;
colset S = product D*P;
colset W= product D*G;
colset R= int;
colset RS= product D*R;
var x: D;
var y: P ;
colset U = union RS +S;
var z:G;
var s:S;
var rs:RS;
var r:R;
var u:U;
Standard priorities
Standard declarations
Monitors
New Page
Binder 0
New Page
1`1
Idle
G
1`"p1"++
1`"p2"
PumpIdle
P
1`"u1"++
1`"u2"++
1`"u3"
getMoney
D
serve
payOk
serveCar
D
Driving
D
pay
Serving
W
payFail
pump
pumpEnabled
S
leave2
ServiceResult
D
served
leave
Ready
D
pumpReady
S
tank
ServiceResult
S
open
Tanking
D
carInPosition
S
Pumping
S
close
pumpRelase
S


VerGasSt1 - Verifier 0.61
File Edit Help
Status Inspector
All
Check It
Check Times
Options
Properties | Value
States | 2626
Edges | 7464
Elapsed Time | 0h:1m:28s:925ms
f.0 >> Check CLK Details ∀□>=0.0(GetMoney1→∀◇>=0.0(PumpEnabled1∨PumpEnabled2)) Edit
f.1 >> Check CLK Details ∀□>=0.0(PumpEnabled1→(∀◇>=0.0Pumping1_1∧¬(∀◇>=0.0Pumping1_2))) Edit
f.2 >> Check CLK Details ∀□>=0.0(PumpEnabled2→(∀◇>=0.0Pumping1_2∧¬(∀◇>=0.0Pumping1_1))) Edit
f.3 >> Check CLK Details ∀□>=0.0(GetMoney2→∀◇>=0.0(PumpEnabled1∨PumpEnabled2)) Edit
f.4 >> Check CLK Details ∀□>=0.0(PumpEnabled1→(∀◇>=0.0Pumping2_1∧¬(∀◇>=0.0Pumping2_2))) Edit
f.5 >> Check CLK Details ∀□>=0.0(PumpEnabled2→(∀◇>=0.0Pumping2_2∧¬(∀◇>=0.0Pumping2_1))) Edit
f.6 >> Check CLK Details ∀□>=0.0(GetMoney3→∀◇>=0.0(PumpEnabled1∨PumpEnabled2)) Edit
f.7 >> Check CLK Details ∀□>=0.0(PumpEnabled1→(∀◇>=0.0Pumping3_1∧¬(∀◇>=0.0Pumping3_2))) Edit
f.8 >> Check CLK Details ∀□>=0.0(PumpEnabled2→(∀◇>=0.0Pumping3_2∧¬(∀◇>=0.0Pumping3_1))) Edit
Memory:

Anais do XX Congresso Brasileiro de Automática
Belo Horizonte, MG, 20 a 24 de Setembro de 2014

VERIFICATION OF AUTOMATED SYSTEMS USING INVARIANTS

Arianna Z. Olivera Salmon*, Pedro M. G. del Foyo†, José R. Silva*

*Design Lab, Department of Mechatronics and Mechanical Systems Engineering, University of Sao Paulo
Av. Professor Mello Moraes, 2231, CEP 05508-970
São Paulo/SP, Brazil

†Mechanical Engineering Department, Federal University of Pernambuco,
Av. Acadêmico Helio Ramos s/n, CEP 50670-901
Recife/PE, Brazil

Emails: arianna@usp.br, pedro.foyo@ufpe.br, reinaldo@usp.br

Abstract— Nowadays, Petri net and this extensions has been used for modeling and verification of complex systems, used as a sound description language. Algorithms derived from this modeling framework can facilitate the analysis and verification of properties. Methods of verification based in invariants are among the most computacionally efficient and allow the verification of other important properties. This work proposes the use of invariants for the verification of desirable properties for an automatic system in the early design phase. Therefore invariants are used to validate requirements assuming they are elicited using UML Diagrams and modeled also in Petri Nets.

Keywords— Invariants, Verification, Petri net.

Resumo— A Rede de Petri e suas extensões são amplamente usadas na modelagem, análise e verificação de sistemas complexos, devido fundamentalmente à expressividade da linguagem de descrição e aos algoritmos derivados deste formalismo, que facilitam a análise e verificação de propriedades. Os métodos de verificação baseados no cálculo de invariantes são considerados entre os algoritmos computacionalmente mais eficientes. Neste trabalho mostraremos mais uma vertente para o uso de invariantes na verificação de propriedade desejáveis em sistemas automatizados. Neste caso, os invariantes serão usados para validar requisitos ainda na fase preliminar do processo de design, assumindo que estes requisitos sejam representados em UML e modelados também em redes de Petri.

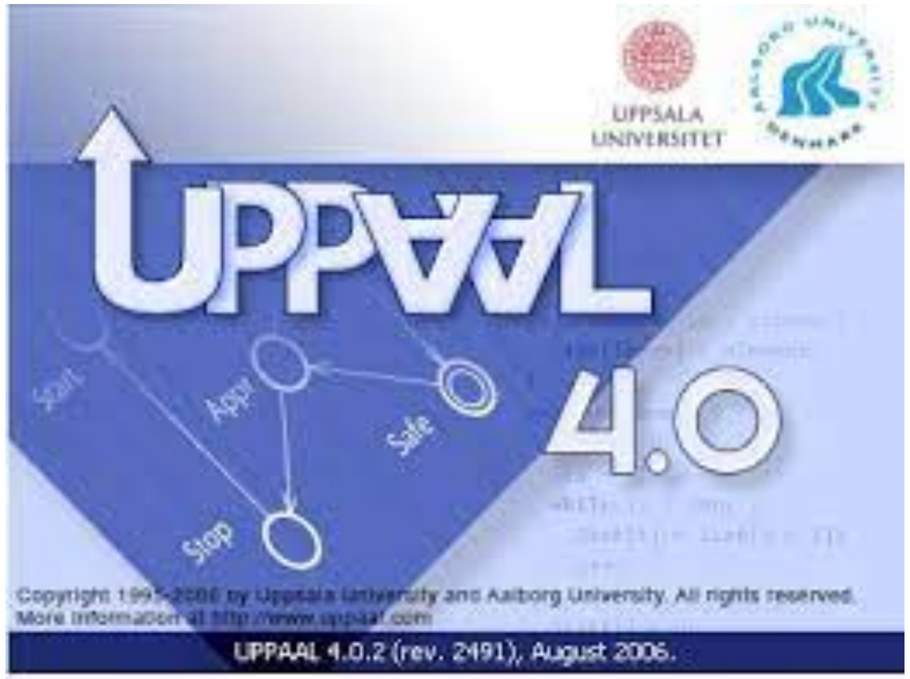
Palavras-chave— Invariantes, análise de requisitos, validação, verificação formal, redes de Petri.

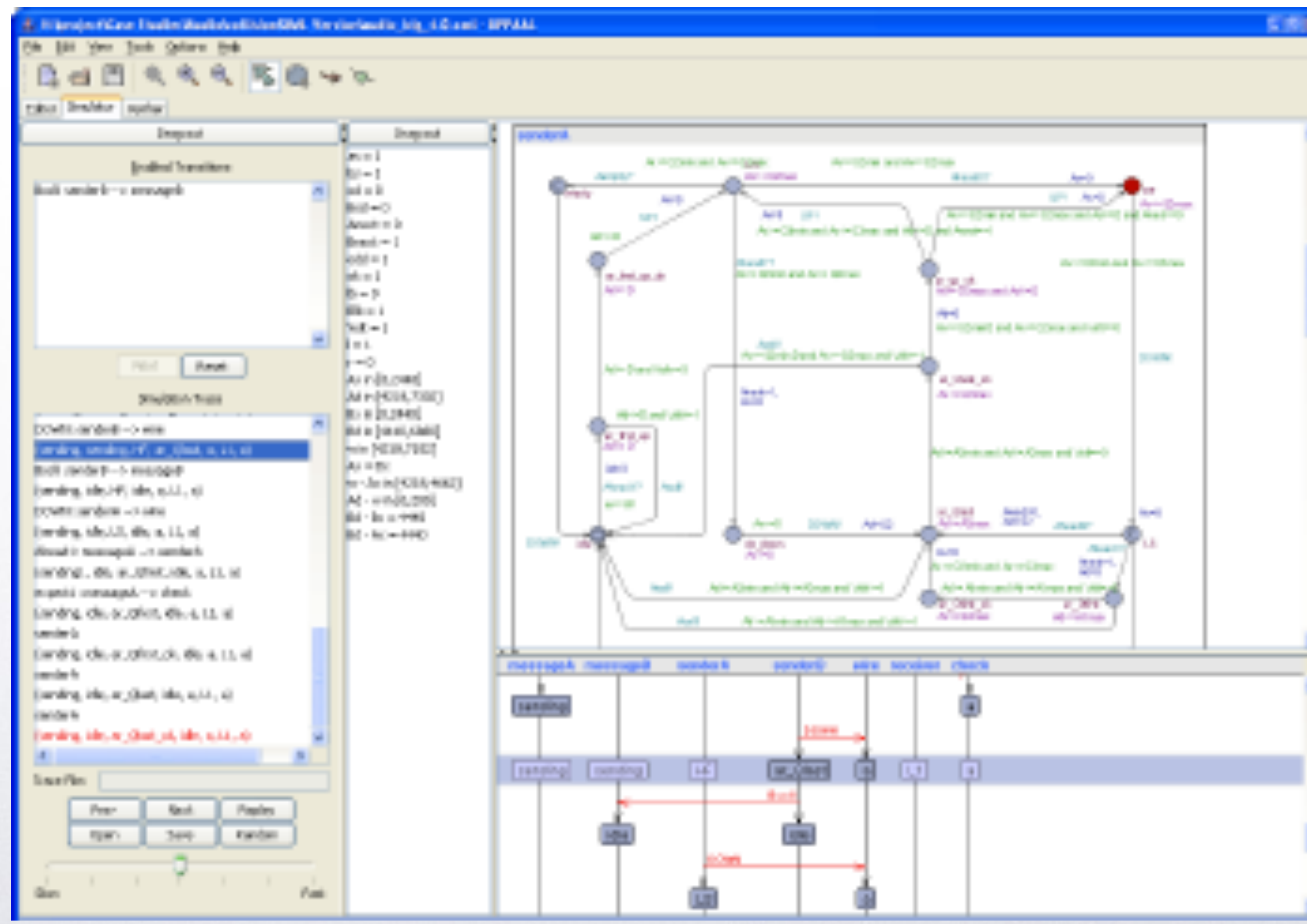
1 Introduction

such as Petri Nets and associated to invariants [1].

State-oriented modeling (SOM) is a very reliable approach to formal verification and produces models correct by construction. However, it implies a state-space analysis, and its generation beforehand or "on the fly". For critical systems, it is the recommended approach, but it is computationally hard.

Who works with that?

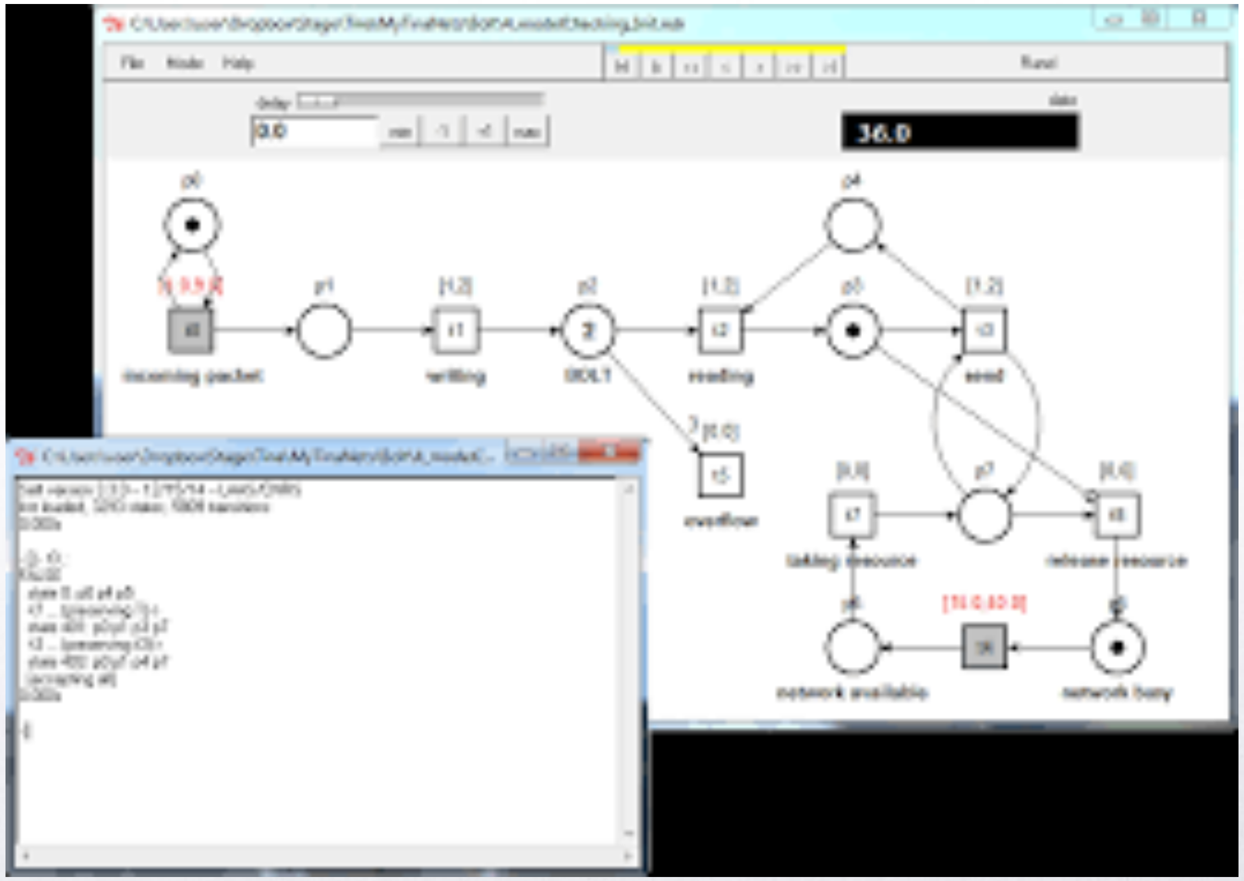




TINA

Who works with that?

LAAS CNRS

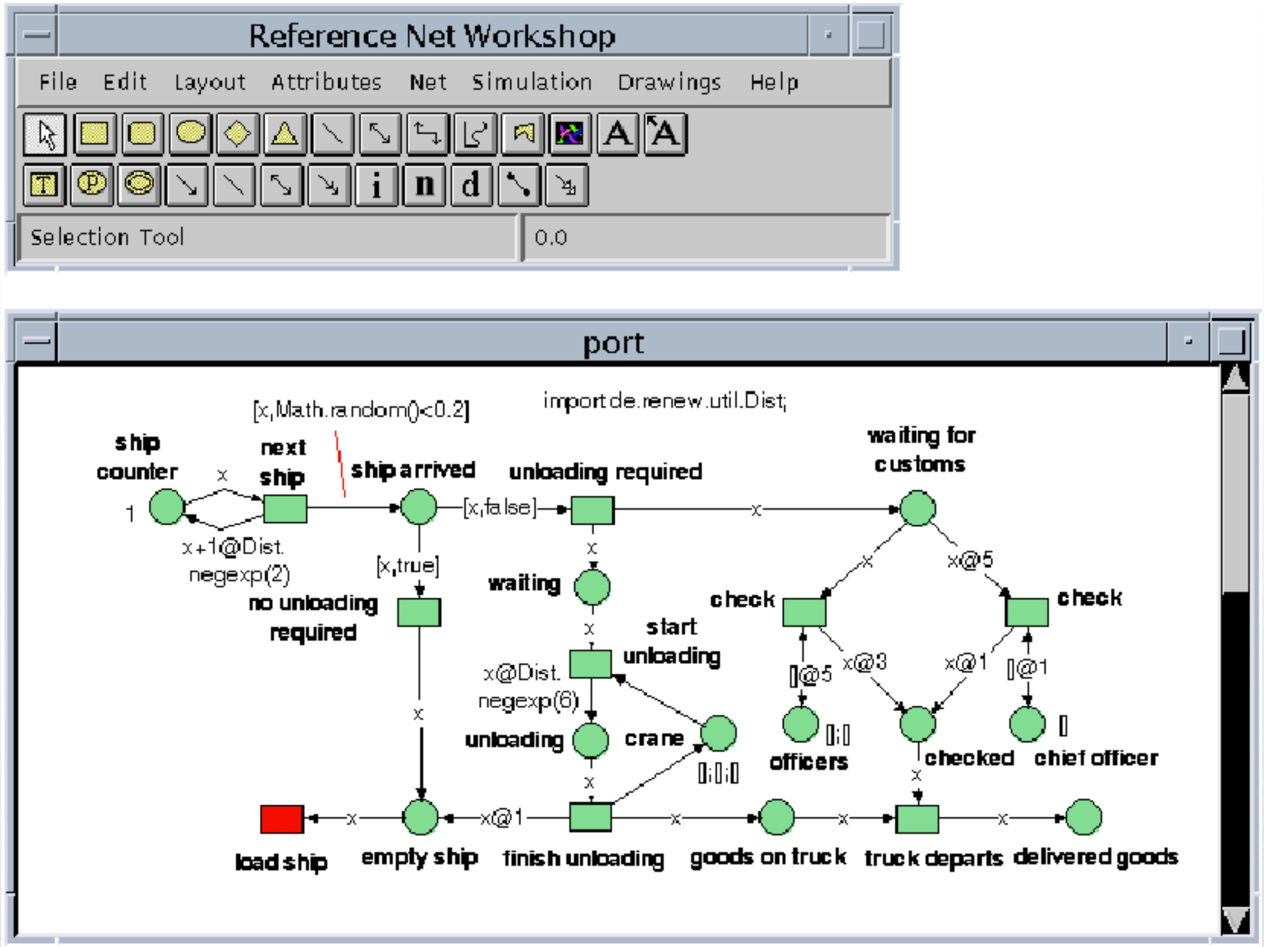


RENEW

Who works with that?

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Theoretical Foundations of Computer Science

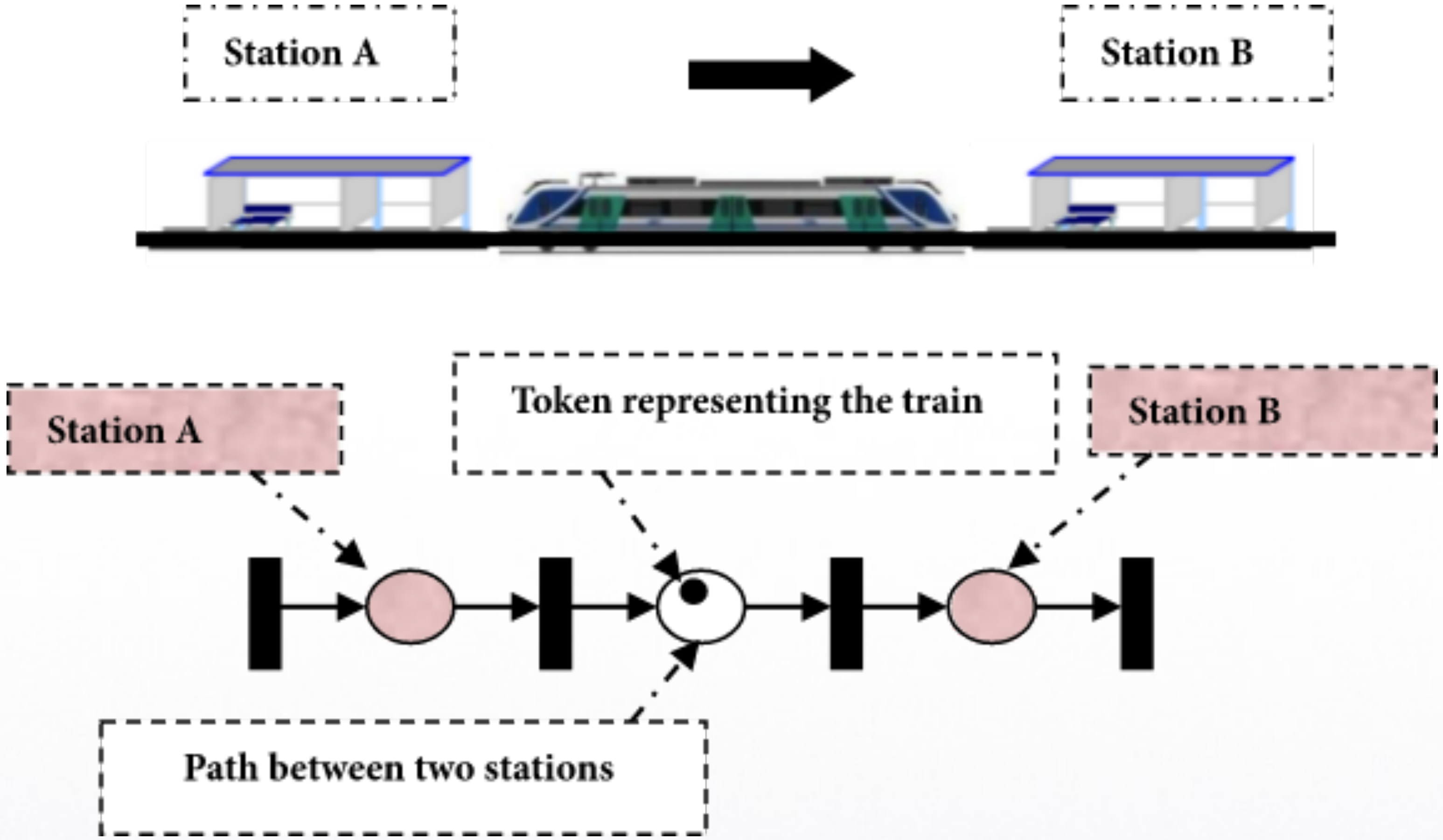


State Space Methods for Coloured Petri Nets

Lars Michael Kristensen

Ph.D. Dissertation

Department of Computer Science
University of Aarhus
Denmark



Event-oriented PN modeling is directed to transactions and workflow.

Na próxima aula vamos ver como aplicar a técnica de building blocks nas redes de alto nível e seguiremos na discussão dos métodos de modelagem de sistemas usando Redes de Petri.