PMR 5020
Metodologia do Projeto de Sistemas

Aula 10: introducing MBSE, features and methods

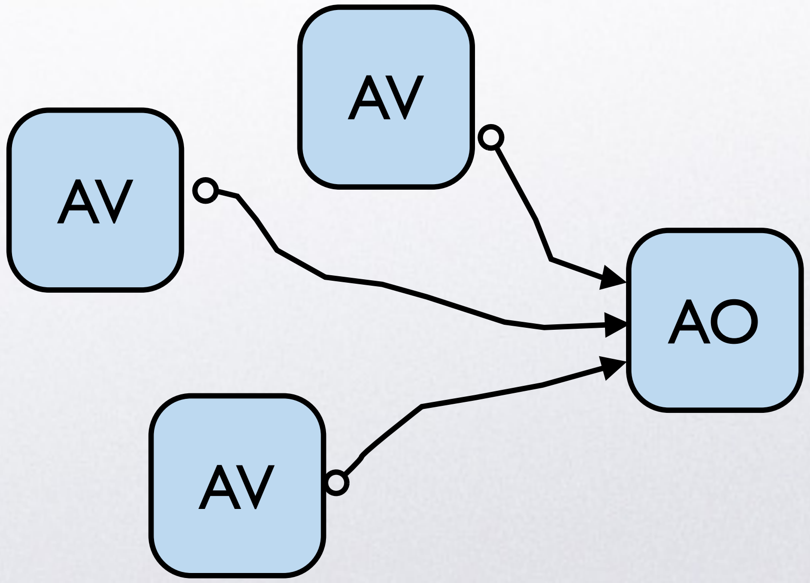
Prof. José Reinaldo Silva
reinaldo@poli.usp.br

Alternativas para o Design de Sistemas

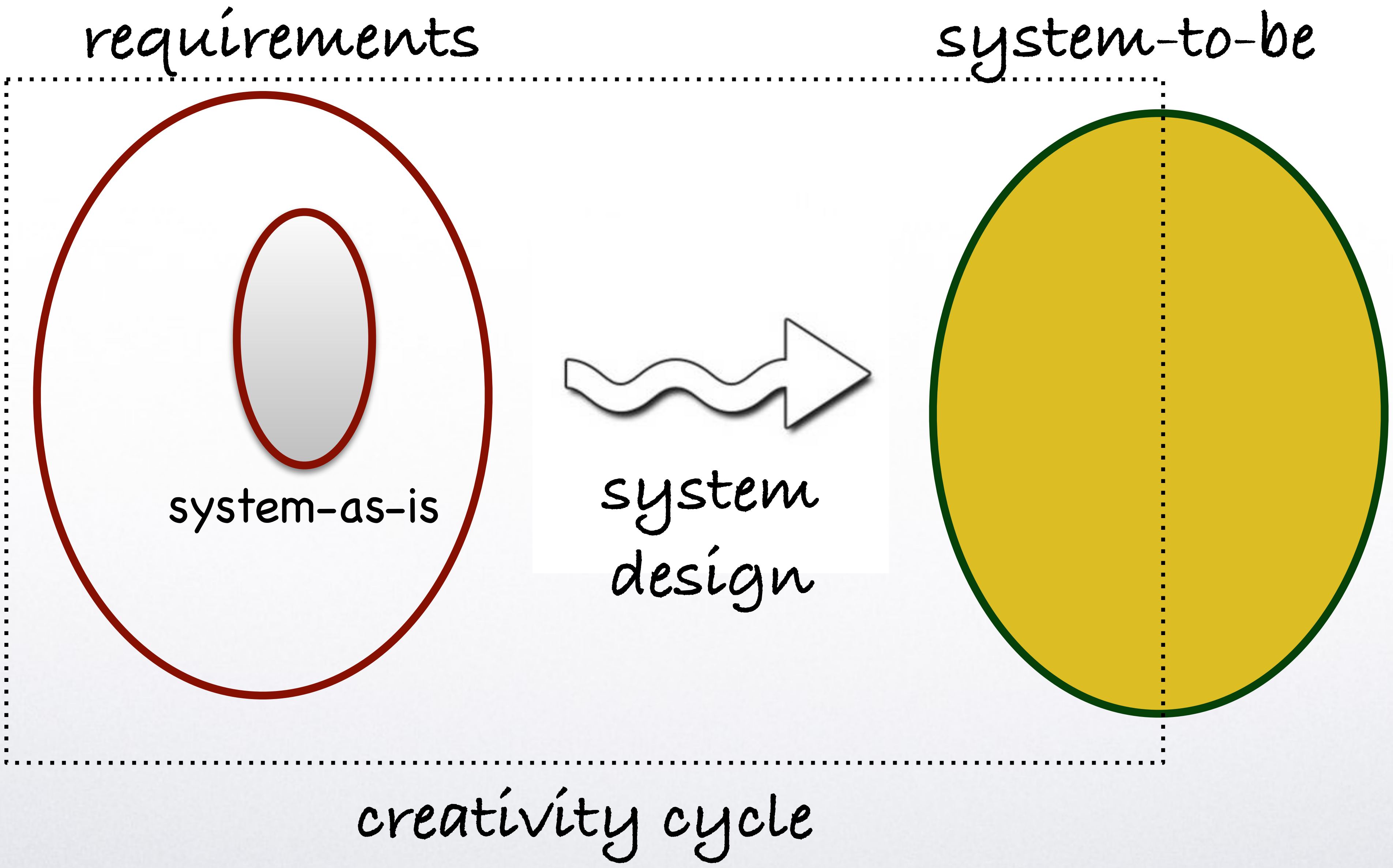
Até aqui vimos as seguintes alternativas para o design de sistemas:

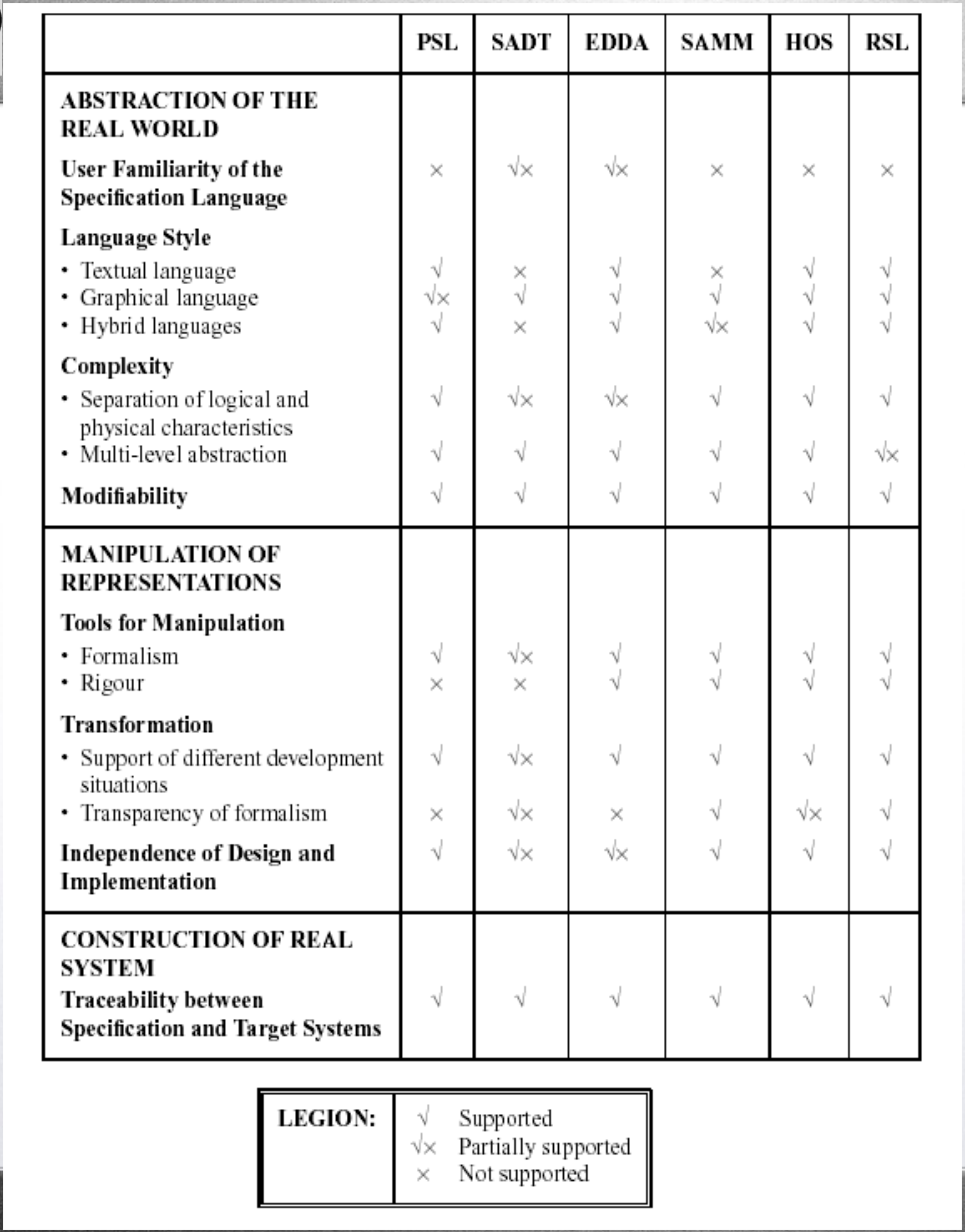
I. investir na fase inicial, na eliciação de requisitos, análise e especificação;
II. a possibilidade de usar várias linguagens na fase de que vem após a especificação;
III. a *esperança* de que a formalização inerente ao processo não seja o motivo de perda de informação ou desvio dos requisitos iniciais;
IV. a possibilidade (especialmente para sistemas automatizados) de sofisticar o controle introduzindo sistemas inteligentes;

Existe de fato um processo de design formal?



The System Design Challenge





	PSL	SADT	EDDA	SAMM	HOS	RSL
ABSTRACTION OF THE REAL WORLD						
User Familiarity of the Specification Language	×	√×	√×	×	×	×
Language Style						
• Textual language	√	×	√	×	√	√
• Graphical language	√×	√	√	√	√	√
• Hybrid languages	√	×	√	√×	√	√
Complexity						
• Separation of logical and physical characteristics	√	√×	√×	√	√	√
• Multi-level abstraction	√	√	√	√	√	√×
Modifiability	√	√	√	√	√	√
MANIPULATION OF REPRESENTATIONS						
Tools for Manipulation						
• Formalism	√	√×	√	√	√	√
• Rigour	×	×	√	√	√	√
Transformation						
• Support of different development situations	√	√×	√	√	√	√
• Transparency of formalism	×	√×	×	√	√×	√
Independence of Design and Implementation	√	√×	√×	√	√	√
CONSTRUCTION OF REAL SYSTEM						
Traceability between Specification and Target Systems	√	√	√	√	√	√

LEGION:		
	√	Supported
	√×	Partially supported
	×	Not supported

PSL - Process Specification Language

SADT - Structured Analysis and Design Technique

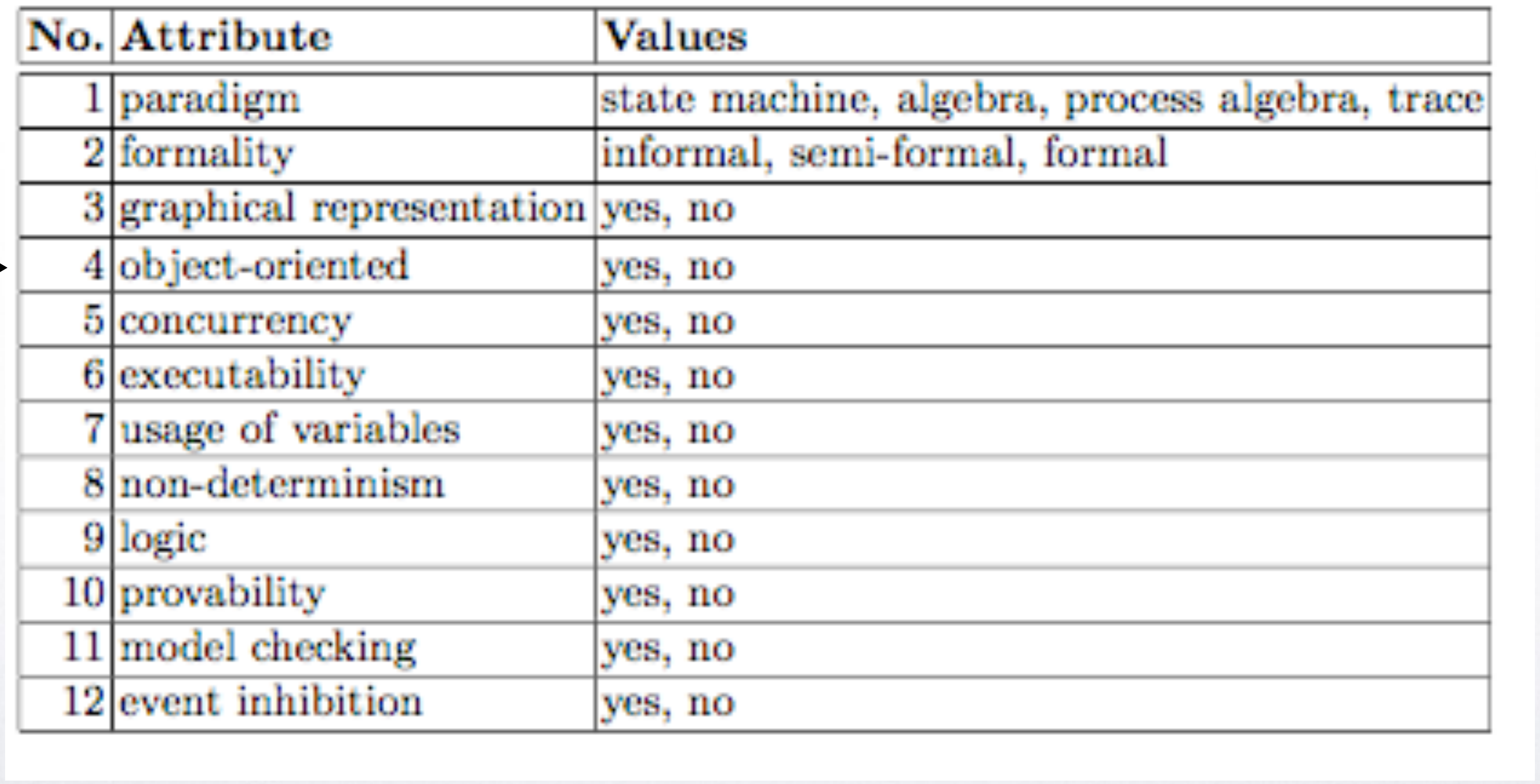
EDDA - Engineering Design and Data Acquisition

SAMM - Systematic Activity Modeling Method

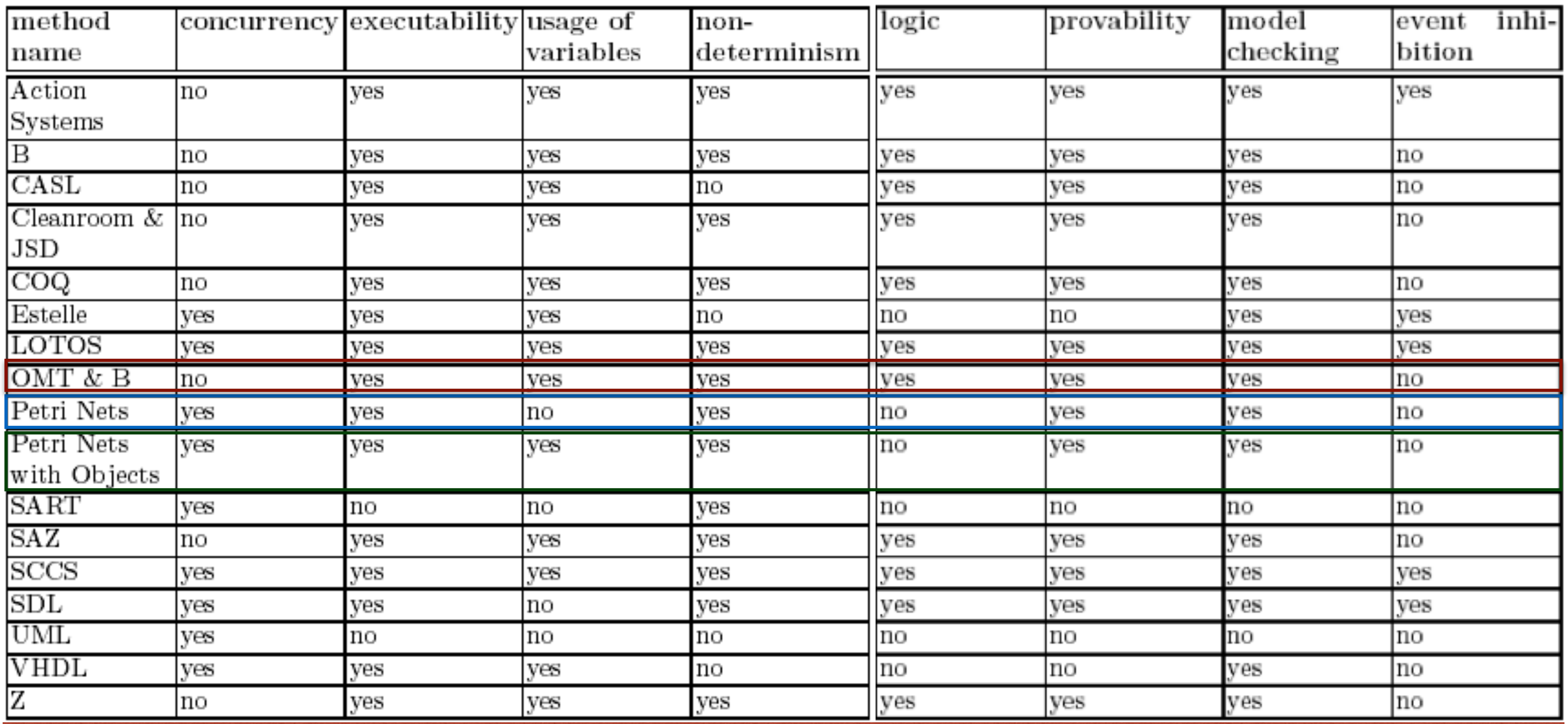
HOS - Hyper Objects Substrate

RSL - Requirements Specification Language

object-orientation



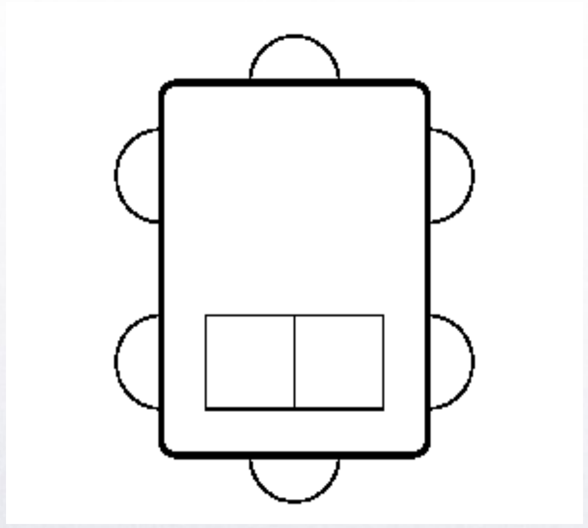
No.	Attribute	Values
1	paradigm	state machine, algebra, process algebra, trace
2	formality	informal, semi-formal, formal
3	graphical representation	yes, no
4	object-oriented	yes, no
5	concurrency	yes, no
6	executability	yes, no
7	usage of variables	yes, no
8	non-determinism	yes, no
9	logic	yes, no
10	provability	yes, no
11	model checking	yes, no
12	event inhibition	yes, no

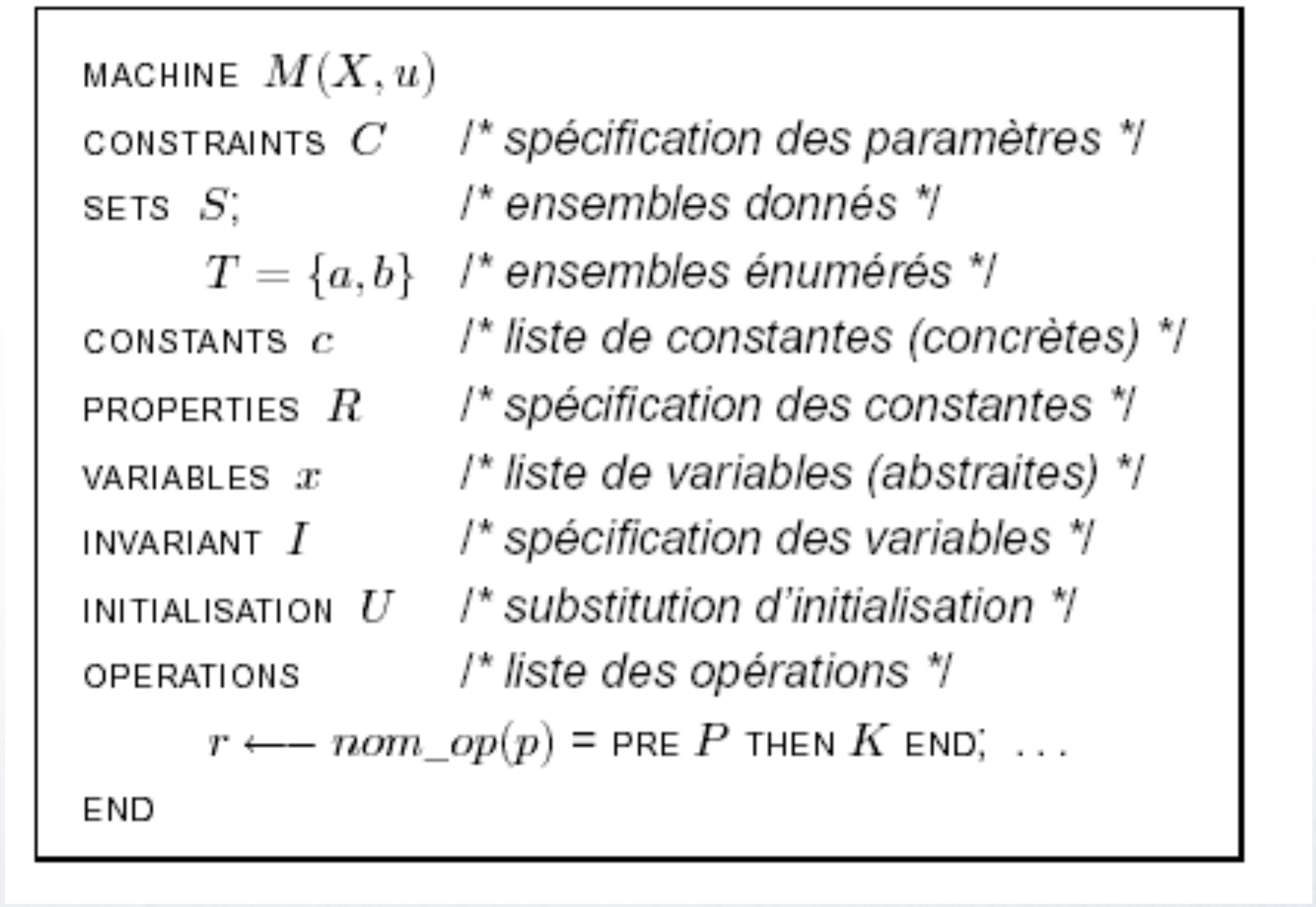


method name	concurrency	executability	usage of variables	non-determinism	logic	provability	model checking	event inhibition
Action Systems	no	yes	yes	yes	yes	yes	yes	yes
B	no	yes	yes	yes	yes	yes	yes	no
CASL	no	yes	yes	no	yes	yes	yes	no
Cleanroom & JSD	no	yes	yes	yes	yes	yes	yes	no
COQ	no	yes	yes	yes	yes	yes	yes	no
Estelle	yes	yes	yes	no	no	no	yes	yes
LOTOS	yes	yes	yes	yes	yes	yes	yes	yes
OMT & B	no	yes	yes	yes	yes	yes	yes	no
Petri Nets	yes	yes	no	yes	no	yes	yes	no
Petri Nets with Objects	yes	yes	yes	yes	no	yes	yes	no
SART	yes	no	no	yes	no	no	no	no
SAZ	no	yes	yes	yes	yes	yes	yes	no
SCCS	yes	yes	yes	yes	yes	yes	yes	yes
SDL	yes	yes	no	yes	yes	yes	yes	yes
UML	yes	no	no	no	no	no	no	no
VHDL	yes	yes	yes	no	no	no	yes	no
Z	no	yes	yes	yes	yes	yes	yes	no

A few words about B

MACHINE …
VARIABLES …
INVARIANT …
INITIALISATION …
OPERATIONS …
END





MACHINE $M(X, u)$	
CONSTRAINTS C	*/* spécification des paramètres */*
SETS $S;$	*/* ensembles donnés */*
$T = \{a, b\}$	*/* ensembles énumérés */*
CONSTANTS c	*/* liste de constantes (concrètes) */*
PROPERTIES R	*/* spécification des constantes */*
VARIABLES x	*/* liste de variables (abstraites) */*
INVARIANT I	*/* spécification des variables */*
INITIALISATION U	*/* substitution d'initialisation */*
OPERATIONS	*/* liste des opérations */*

$r \longleftarrow nom\_op(p)$ = PRE P THEN K END; $\ldots$

END

Estratégia de modelagem

A base para a modelagem de um sistema em B é a máquina abstrata. Em outras palavras, a modelagem de um sistema é uma associação lógica de máquinas abstratas onde:

- vale o princípio da composicionalidade, isto é, uma associação de máquinas abstratas é também uma máquina abstrata;
- Cada máquina encapsula sua capacidade na forma de operações que pode realizar e se assemelha a um objeto (com herança simples e agregação);
- Existe uma arquitetura hierárquica entre as máquinas abstratas

Base para a modelagem

A base para a modelagem é a teoria de conjuntos onde se destacam os conjuntos e as expressões.

$S_1 \times S_2$	Produto cartesiano
$\mathbb{P}(S)$	Conjunto das partes
$\{\, x \mid P \,\}$	Regra de composição interna
BIG	Um conjunto infinito

x	Variável
$[x := E_1]E_2$	Substituição em uma expressão
(E_1, E_2)	Um par de expressões
$\text{choice}(S)$	Função de escolha
S	Um conjunto

Base do formalismo lógico

Lógica de predicados

$P \wedge Q,\ P \Rightarrow Q,\ \neg P$	Predicados base
$\forall x \cdot P$	Quantificador universal
$[x := E]\, P$	Substituição em um predicado

Outros predicados envolvendo conjuntos

$x \in S$	Relação de pertinência
$E_1 = E_2$	igualdade

As demais relações se formam como uma combinação das anteriores

$P \vee Q,\ \exists x \cdot P,\ \ x \notin S$, etc.

Exemplo: um elevador

• uma porta em cada andar
• os comandos vindos do interior ou da parte externa são iguais
• não existem falhas
• o número máximo de andares é max_andar > 0

As operações sobre o elevador são:

• abrir e fechar a porta

• chamar o elevador

• deslocar o elevador

Propriedades do elevador

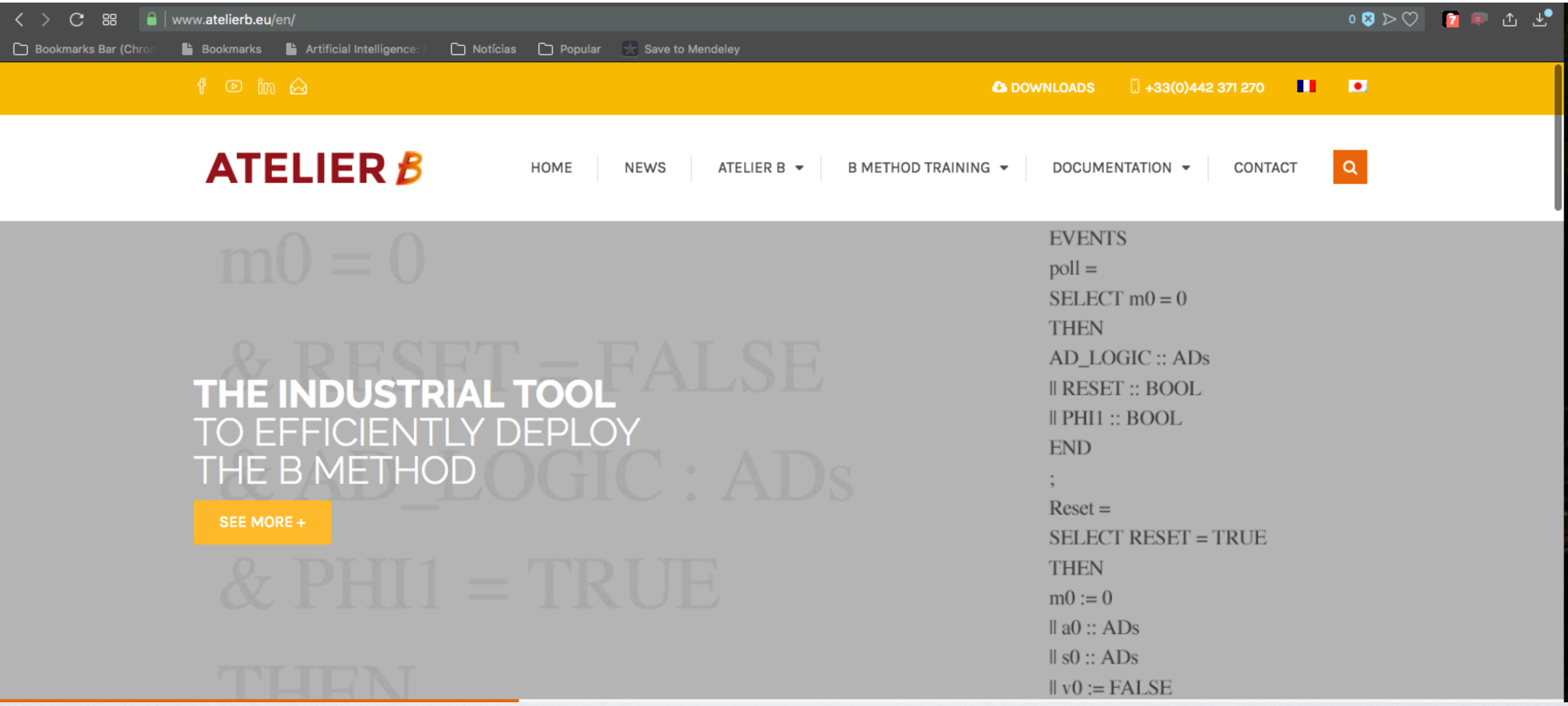
- o elevador se move no limite dos andares 0 (térreo) e max_andar;
- a porta de um dado andar só se abre se o elevador estiver neste mesmo andar;
- cada chamada deve ser atendida dentro de um certo intervalo de tempo t_{max};
- se o elevador chega a um andar a chamada vinda deste andar é considerada atendida.

Máquina abstrata

MACHINE: Elevador;
SETS : ESTADO = {parado, movimentando};
CONSTANTES : max_andar, ANDARES;
PROPRIEDADES: max_andar $\in$ NAT* $\wedge$ ANDARES = 0..max_andar;
VARIÁVEIS : chamada, pos, porta_aberta $\in$ ANDARES
estado $\in$ ESTADO;
INVARIANTES: (estado=parado $\Rightarrow$ chamada = porta_aberta $\vee$
estado=movimentando $\Rightarrow$ chamada $\neq$ porta_aberta) ;

OPERADORES: (porta_aberta=max_andar) $\Rightarrow$ (destino= max_andar – 1)
(porta_aberta=0) $\Rightarrow$ (destino= 1);
[(porta_aberta > 0) $\wedge$ (porta_aberta < max_andar)] $\Rightarrow$ [(destino=porta_aberta + 1) $\vee$
(destino=porta_aberta – 1)]

www.atelierb.eu


www.atelierb.eu/en/

Bookmarks Bar (Chro | Bookmarks | Artificial Intelligence | Notícias | Popular | Save to Mendeley

DOWNLOADS | +33(0)442 371 270

ATELIER B

HOME | NEWS | ATELIER B | B METHOD TRAINING | DOCUMENTATION | CONTACT

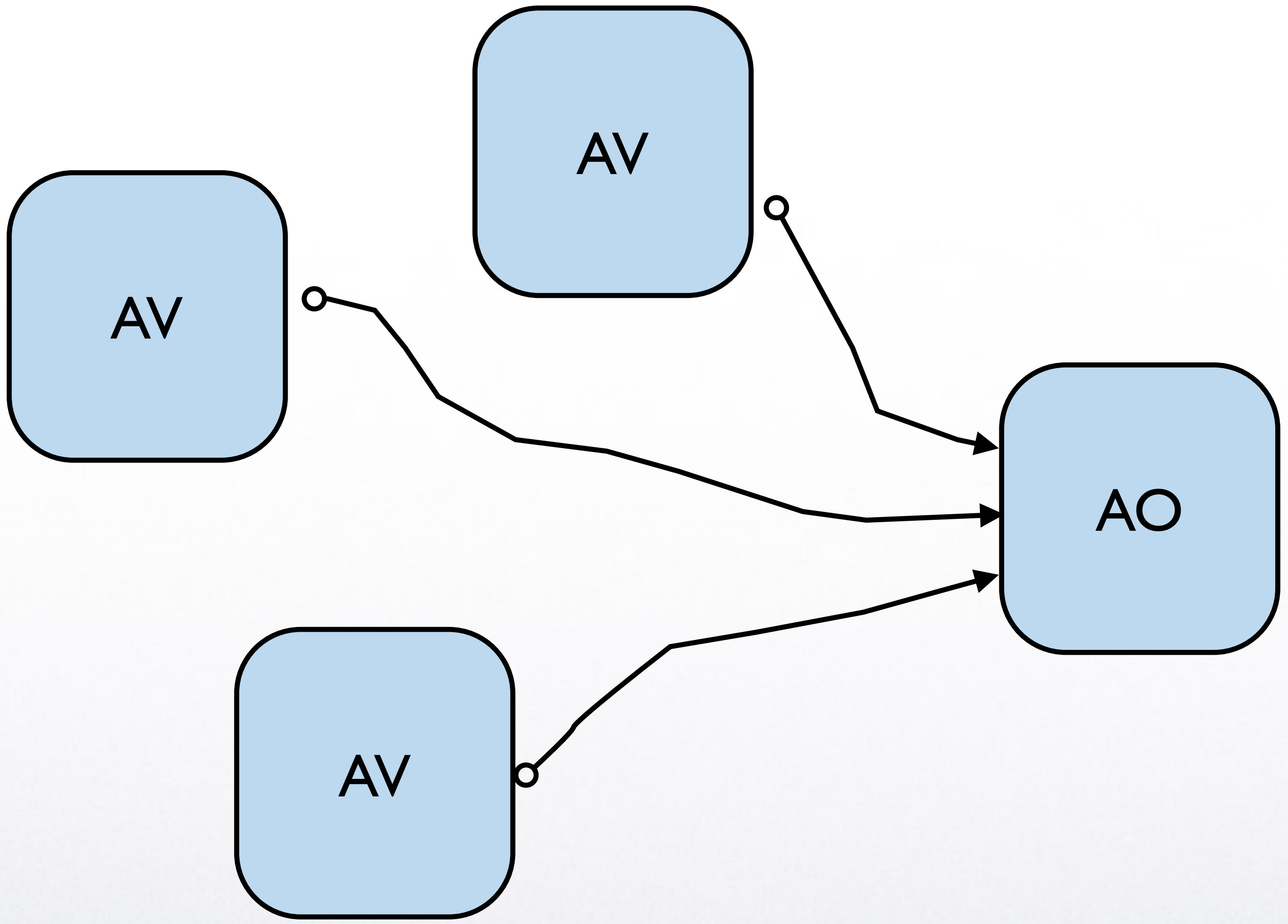
m0 = 0
& RESET = FALSE
& AD_LOGIC : ADs
& PHI1 = TRUE
THEN

THE INDUSTRIAL TOOL
TO EFFICIENTLY DEPLOY
THE B METHOD

SEE MORE +

```
EVENTS
poll =
SELECT m0 = 0
THEN
AD_LOGIC :: ADs
|| RESET :: BOOL
|| PHI1 :: BOOL
END
;
Reset =
SELECT RESET = TRUE
THEN
m0 := 0
|| a0 :: ADs
|| s0 :: ADs
|| v0 := FALSE
```

Functional or not functional ?



A Polymorphic view of functionality (C. Lucena, J. R. Silva)

Object Oriented Design

- o sistema é composto por um conjunto de objetos
- o estado do sistema é dado pelos atributos de todas as instâncias de objeto
- uma transição no sistema se dá através de mensagens que por sua vez dispara um ou mais métodos.

Systems Design Phases

Planning

Feasibility - id. resources - impact analysis

Requirements

Elicitation - Analysis - Validate - Formalize

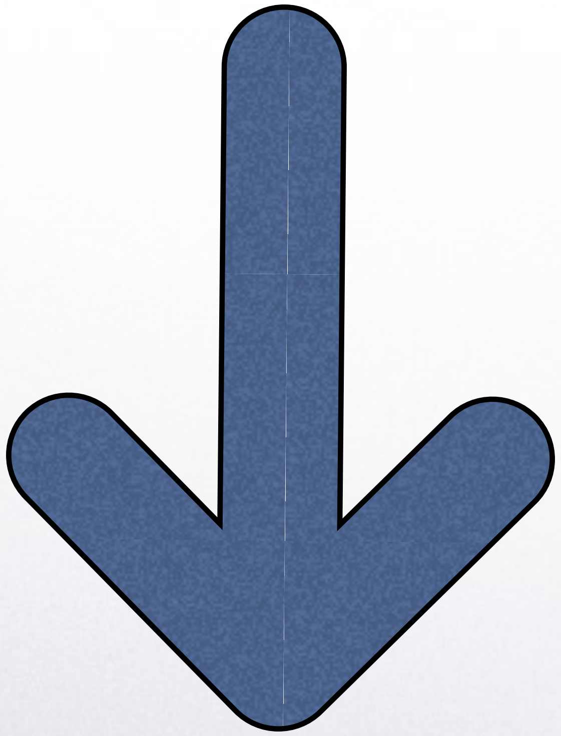
Design

Transfer formal specs from Req. to Implementation

Prototyping

Implementation

Develop-Test-Deploy-Operate-Maintain-Destroy



unified notation

The big challenge in today's class is to answer the question: how could be scale the concepts discussed so far to large and complex (automated) systems, indeed, how could we apply all this knowledge - claimed to be effective - to real life projects?

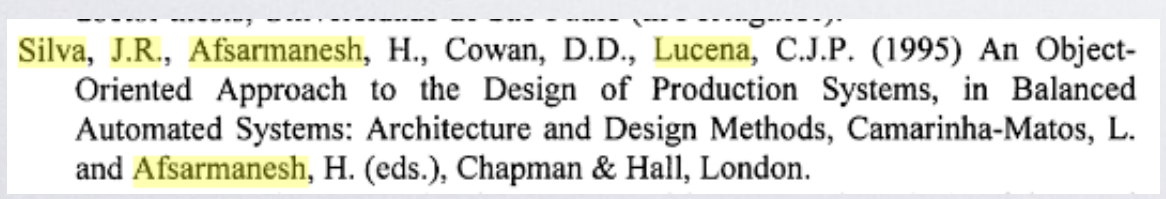
→ Model Driven Engineering

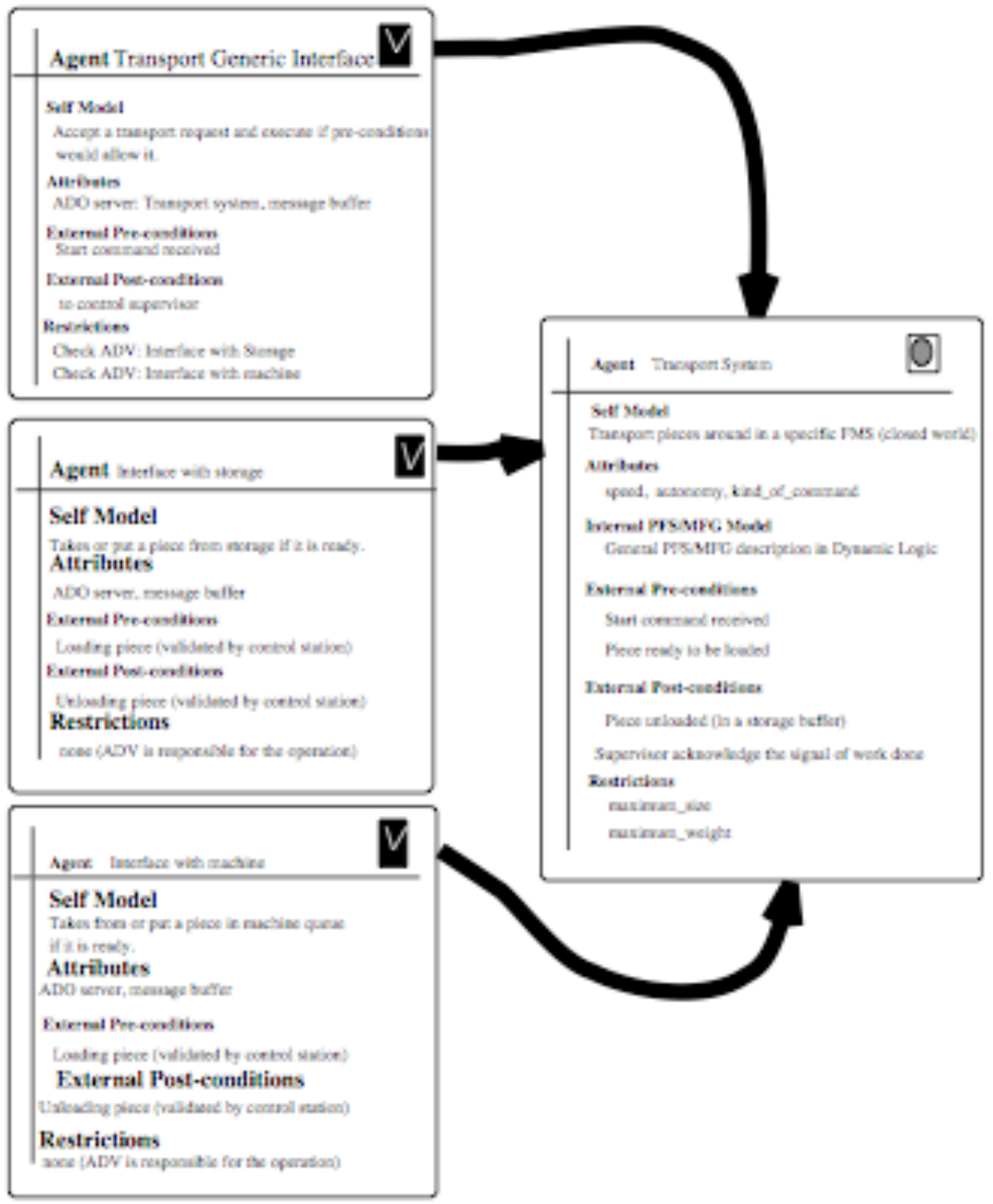
… which concepts (formal or not) are we talking abou?

- Separation of concerns (extracted from the object-oriented appaoch);
- Anticipate formalization (specially in the requirements phase);
- Distributed focus on abstraction, treceability and communication (from req., design and implem.);
- Good selection for a formal representation (language);
- Using systems of systems;
- Introduction of service-orientation and of product-service.

Object interface-service relationship

Separation of concerns
(herdado da abordagem orientada a objetos)


Silva, J.R., Afsarmanesh, H., Cowan, D.D., Lucena, C.J.P. (1995) An Object-Oriented Approach to the Design of Production Systems, in Balanced Automated Systems: Architecture and Design Methods, Camarinha-Matos, L. and Afsarmanesh, H. (eds.), Chapman & Hall, London.


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

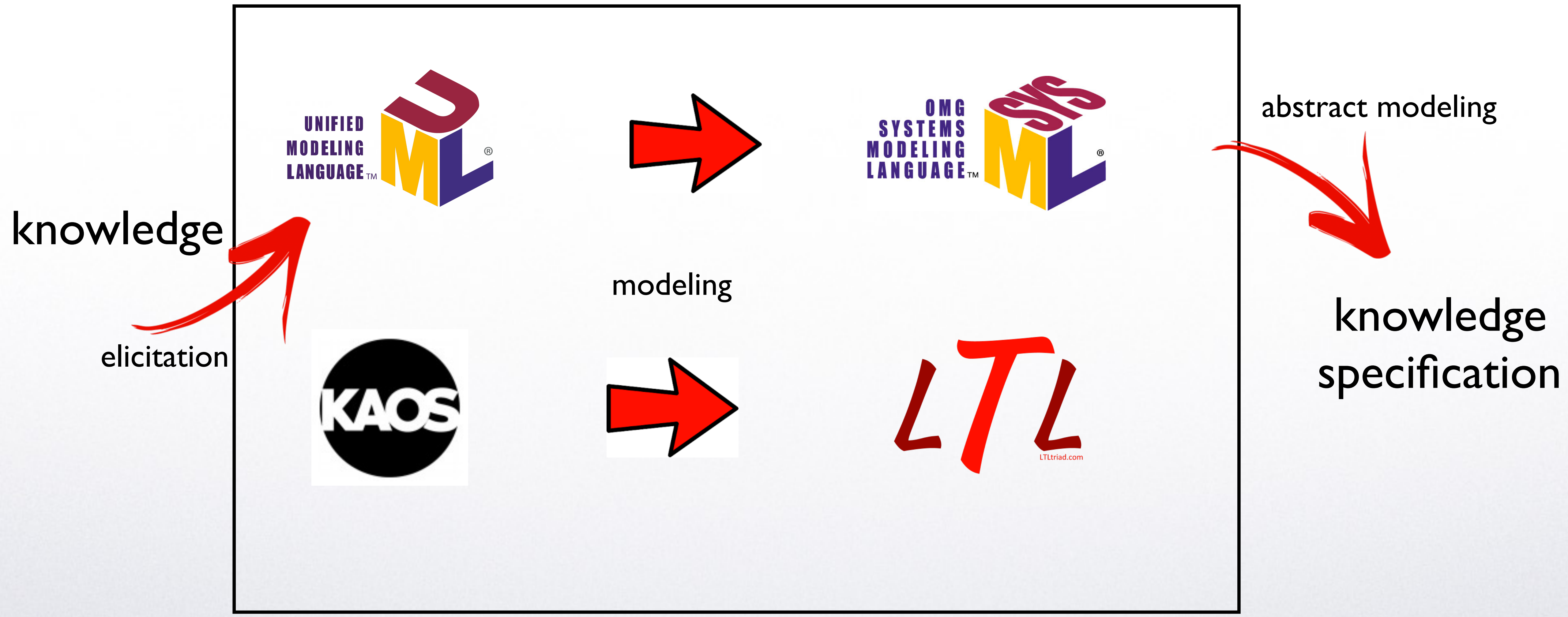
Self Model
Takes from or put a piece in machine queue if it is ready.
Attributes
ADO server, message buffer
External Pre-conditions
Loading piece (validated by control station)
External Post-conditions
Unloading piece (validated by control station)
Restrictions
none (ADV is responsible for the operation)

Agent Transport System

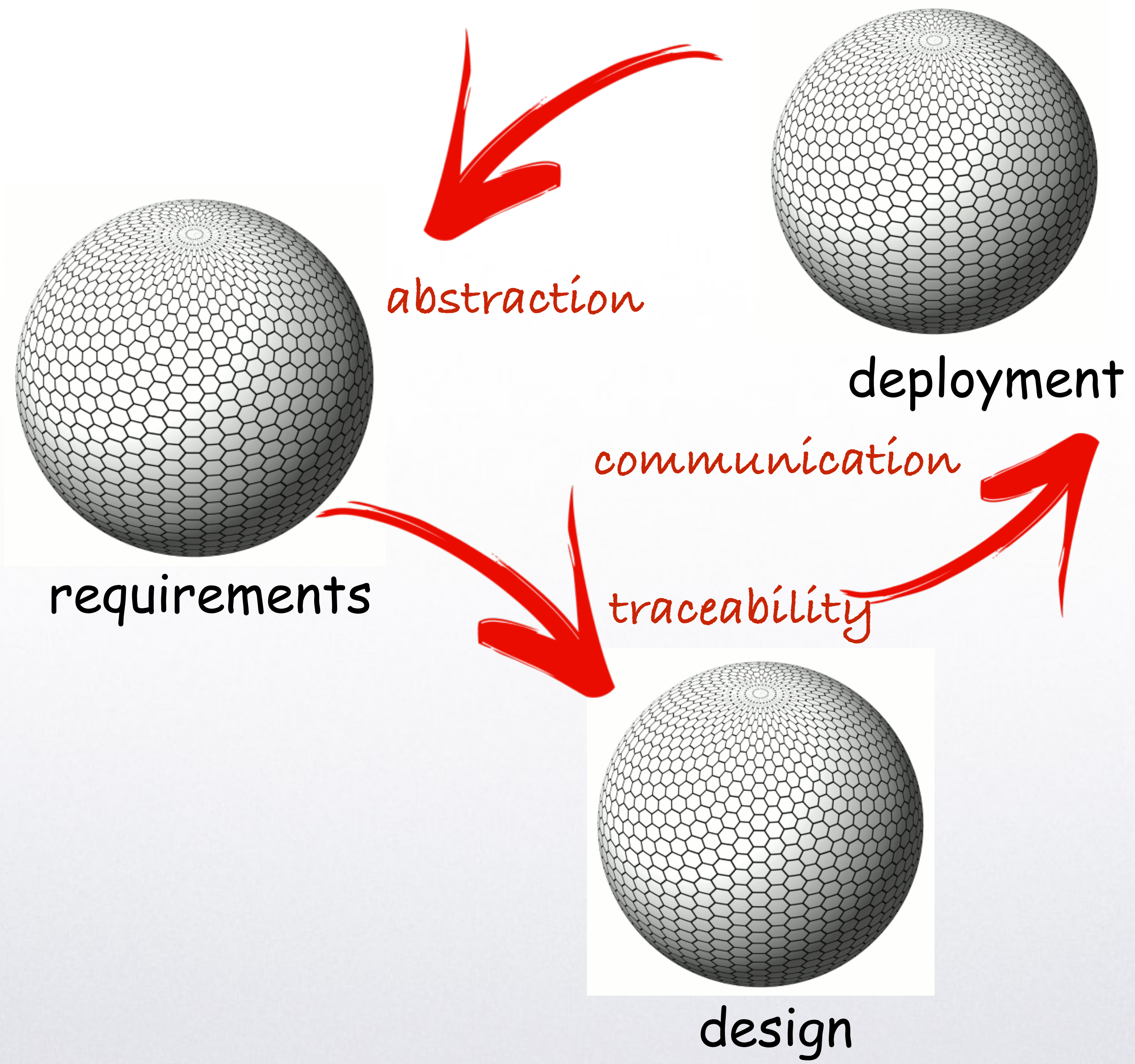
Self Model
Transport pieces around in a specific FMS (closed world)
Attributes
speed, autonomy, kind_of_command
Internal PFS/MFG Model
General PFS/MFG description in Dynamic Logic
External Pre-conditions
Start command received
Piece ready to be loaded
External Post-conditions
Piece unloaded (in a storage buffer)
Supervisor acknowledge the signal of work done
Restrictions
maximum_size
maximum_weight

Anticipate formalization (specially in requirements)

Requirements Engineering

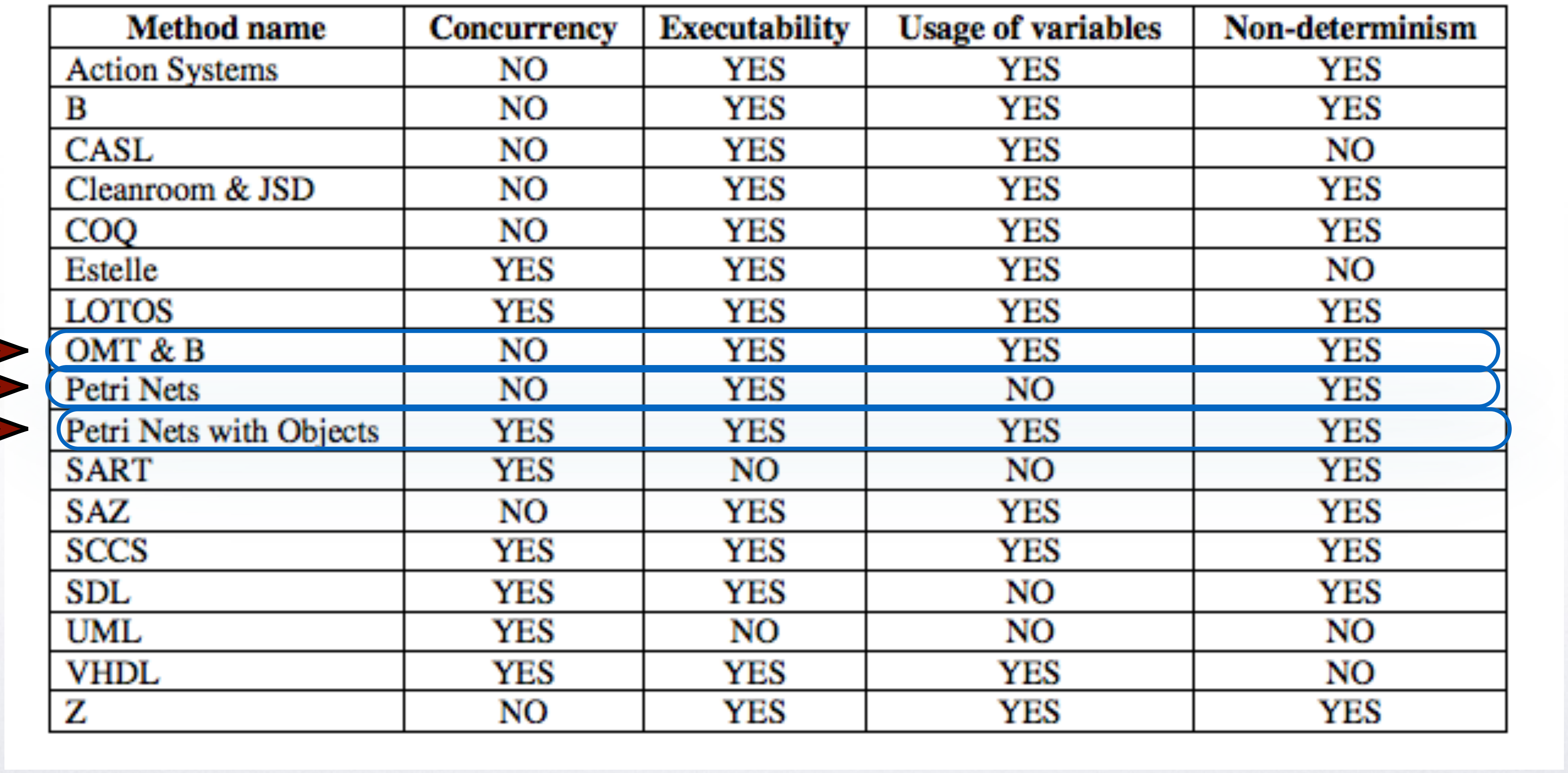


Distributed focus on abstraction, traceability and communication



Escolha de uma boa linguagem de design (formal)

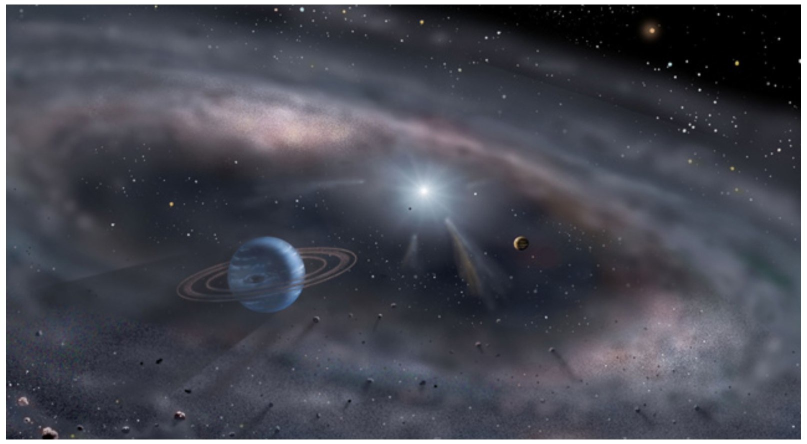
Available Languages

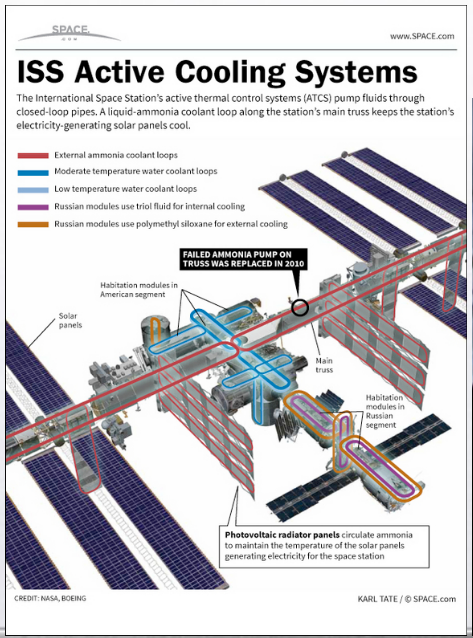


Method name	Concurrency	Executability	Usage of variables	Non-determinism
Action Systems	NO	YES	YES	YES
B	NO	YES	YES	YES
CASL	NO	YES	YES	NO
Cleanroom & JSD	NO	YES	YES	YES
COQ	NO	YES	YES	YES
Estelle	YES	YES	YES	NO
LOTOS	YES	YES	YES	YES
OMT & B	NO	YES	YES	YES
Petri Nets	NO	YES	NO	YES
Petri Nets with Objects	YES	YES	YES	YES
SART	YES	NO	NO	YES
SAZ	NO	YES	YES	YES
SCCS	YES	YES	YES	YES
SDL	YES	YES	NO	YES
UML	YES	NO	NO	NO
VHDL	YES	YES	YES	NO
Z	NO	YES	YES	YES

The System of Systems Challenge

A practical obstacle to the formalization of design is the effectiveness of this approach in practice, specially for complex systems. Generally, formal approaches do not "fit" the complexity of large systems (of systems).




SPACE.com www.SPACE.com

ISS Active Cooling Systems

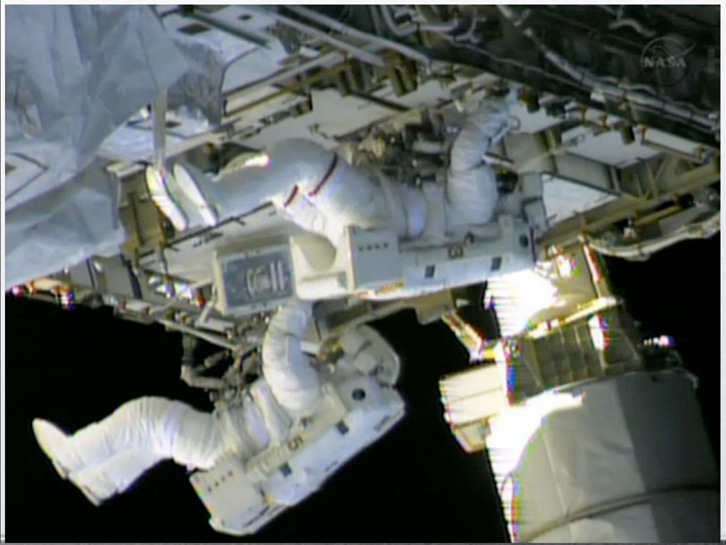
The International Space Station's active thermal control systems (ATCS) pump fluids through closed-loop pipes. A liquid-ammonia coolant loop along the station's main truss keeps the station's electricity-generating solar panels cool.

🟥 External ammonia coolant loops
🟦 Moderate temperature water coolant loops
🟦 Low temperature water coolant loops
🟪 Russian modules use triol fluid for internal cooling
🟧 Russian modules use polymethyl siloxane for external cooling

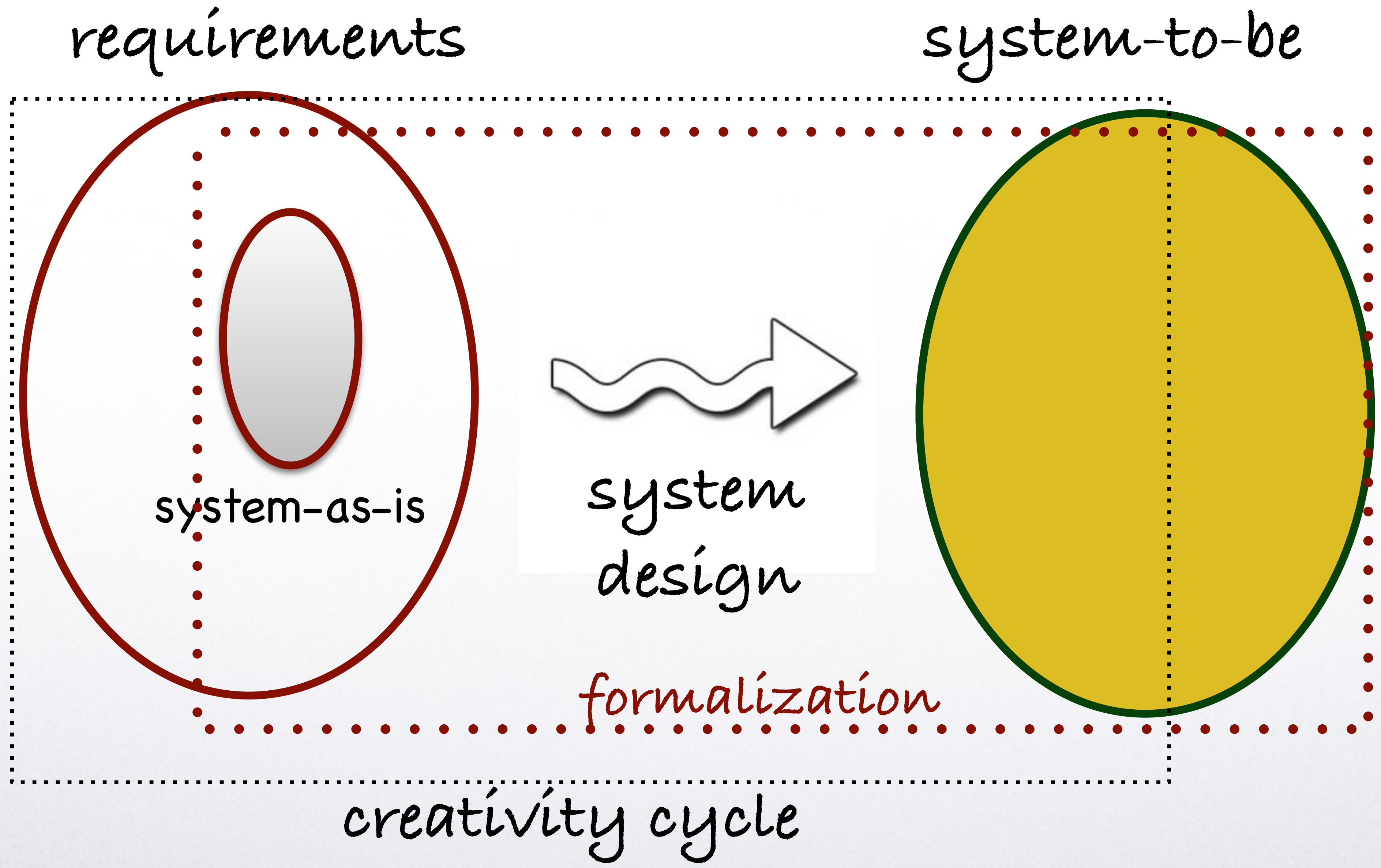
FAILED AMMONIA PUMP ON TRUSS WAS REPLACED IN 2010
Habitation modules in American segment
Solar panels
Main truss
Habitation modules in Russian segment

Photovoltaic radiator panels circulate ammonia to maintain the temperature of the solar panels generating electricity for the space station

CREDIT: NASA, BOEING KARL TATE / © SPACE.com

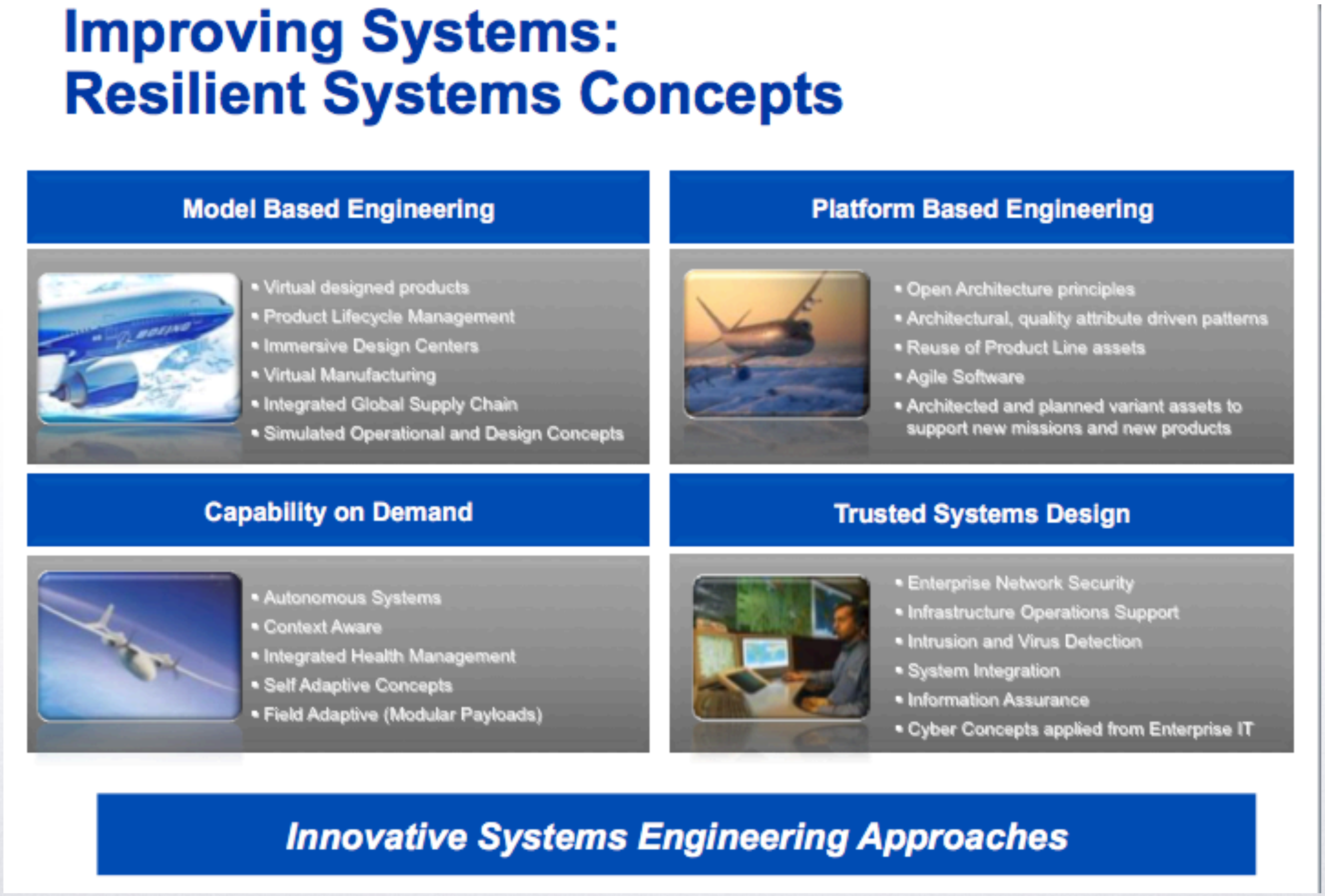


The System Design Challenge



The big challenge in today's class is to answer the question: how could be scale the concepts discussed so far to large and complex (automated) systems, indeed, how could we apply all this knowledge - claimed to be effective - to real life projects?

→ Model Driven Engineering



Improving Systems: Resilient Systems Concepts

Model Based Engineering

- Virtual designed products
- Product Lifecycle Management
- Immersive Design Centers
- Virtual Manufacturing
- Integrated Global Supply Chain
- Simulated Operational and Design Concepts

Platform Based Engineering

- Open Architecture principles
- Architectural, quality attribute driven patterns
- Reuse of Product Line assets
- Agile Software
- Architected and planned variant assets to support new missions and new products

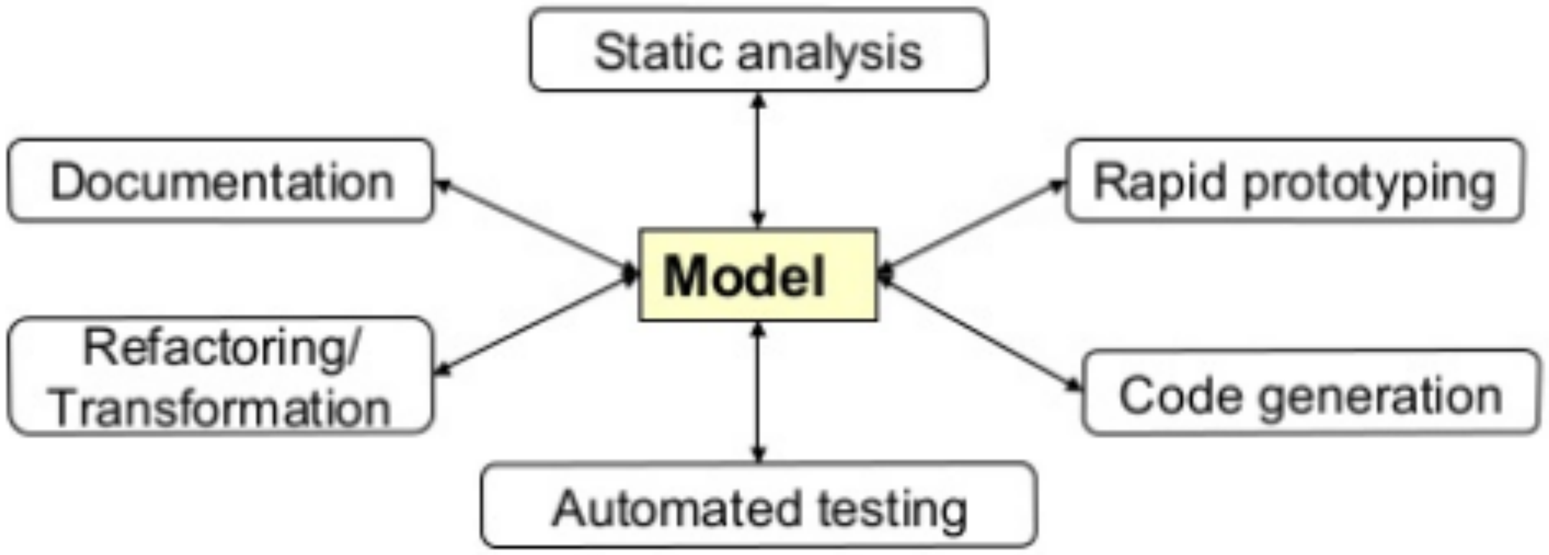
Capability on Demand

- Autonomous Systems
- Context Aware
- Integrated Health Management
- Self Adaptive Concepts
- Field Adaptive (Modular Payloads)

Trusted Systems Design

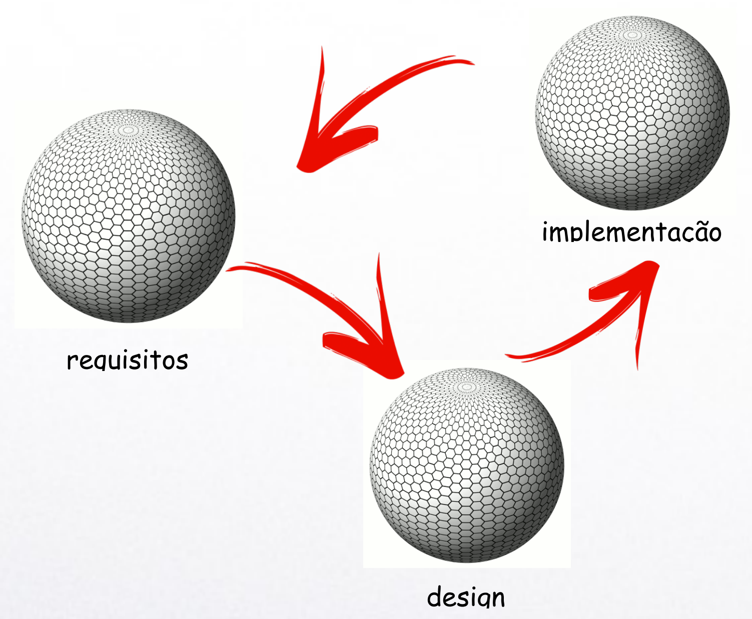
- Enterprise Network Security
- Infrastructure Operations Support
- Intrusion and Virus Detection
- System Integration
- Information Assurance
- Cyber Concepts applied from Enterprise IT

Innovative Systems Engineering Approaches

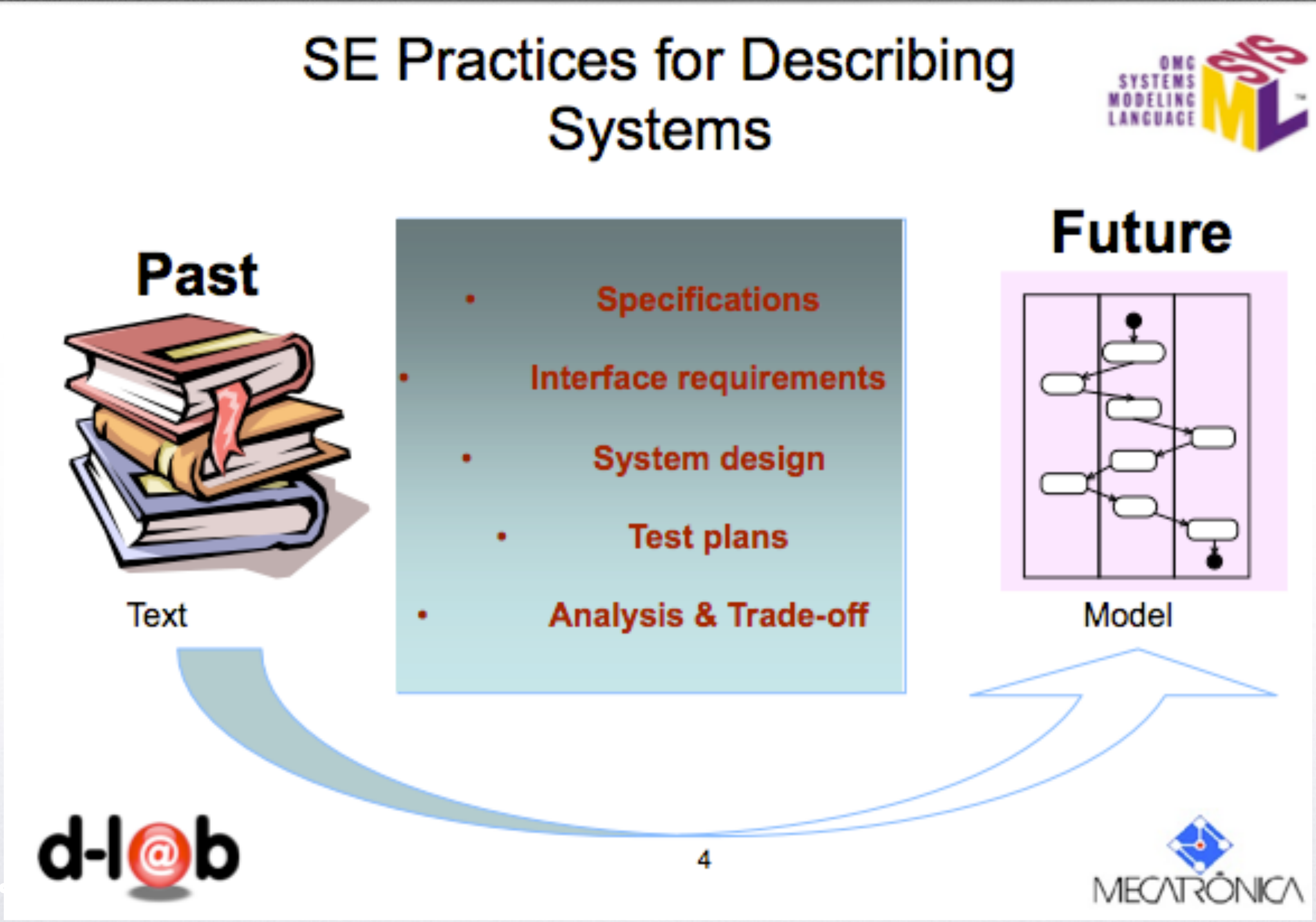


- Related terms
  - Model Driven Engineering (MDE),
  - Model Driven [Software] Development (MDD/MDSD),
  - Model Driven Architecture (MDA)
  - Model Integrated Computing (MIC)

Revendo o problema "prático" do Systems Design



O design "ideal" persegue arduamente correção e completeza, o que só se consegue com a formalização. No entanto, quanto mais formal o sistema mais difícil é a comunicação entre as grandes "esferas": requisito (o que tem que ser feito e porque), o design (como se consegue o sistema e seus objetivos de forma direta e com menor custo e maior eficiência), e a implementação (como fazer e como eliminar o sistema e como interagir de forma eficiente com o usuário final).



SE Practices for Describing Systems

Past

Text

- **Specifications**
- **Interface requirements**
- **System design**
- **Test plans**
- **Analysis & Trade-off**

Future

Model

4

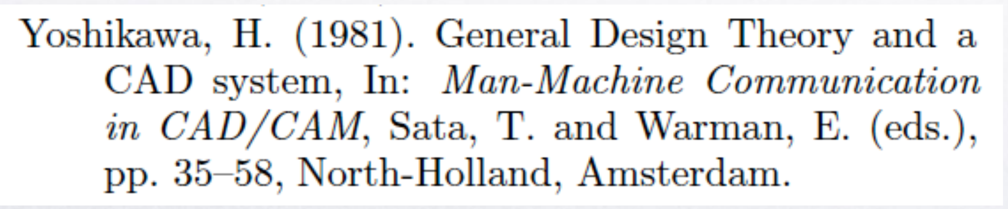
Portanto, o grande (falso) dilema é formalização versus boas práticas. Porque falso? Porque o design não vive sem os dois.

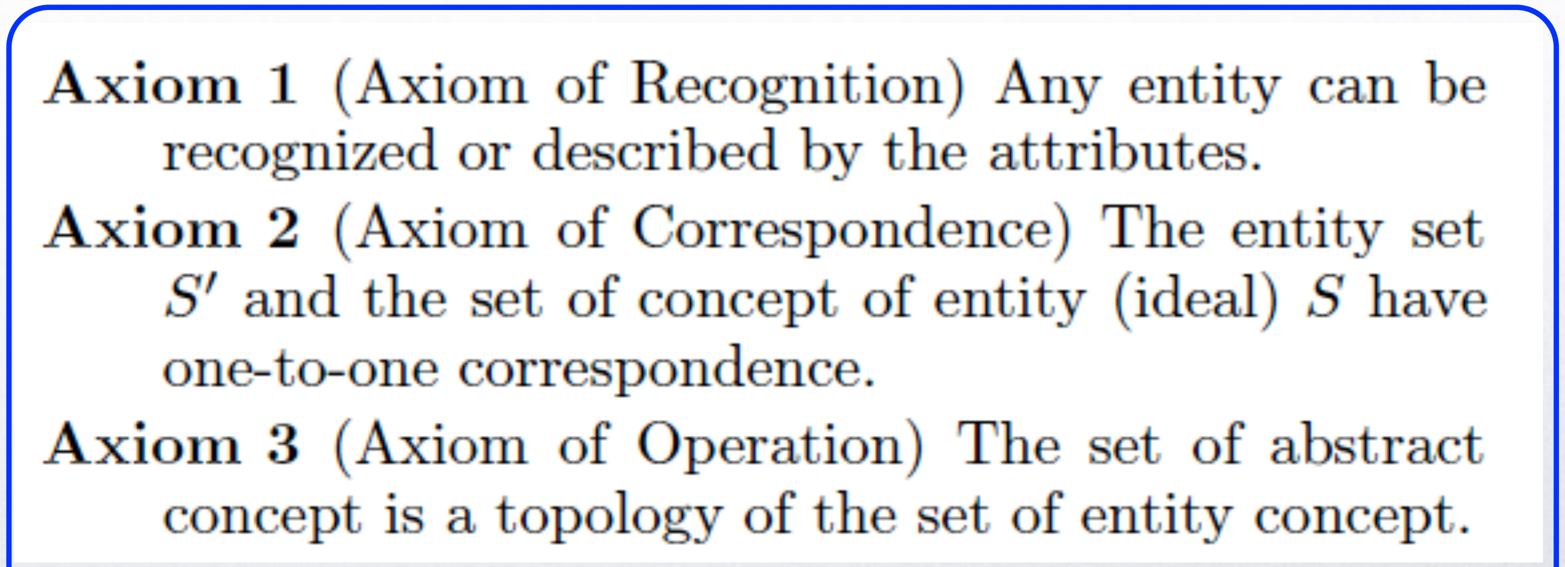
A busca por uma Teoria Geral do Design

A discussão sobre a formalização do processo de design começou em 1981com a proposta de Hiroyuki Yoshikawa, e foi logo em seguida ampliada por Tetsuo Tomiyama, seu orientado. A polemica perdura até hoje e varia de alegações ao arcabouço teórico, à abordagem conceitual, até a perspectiva de aplicação.

Hiroyuki Yoshikawa

Tetsuo Tomiyama

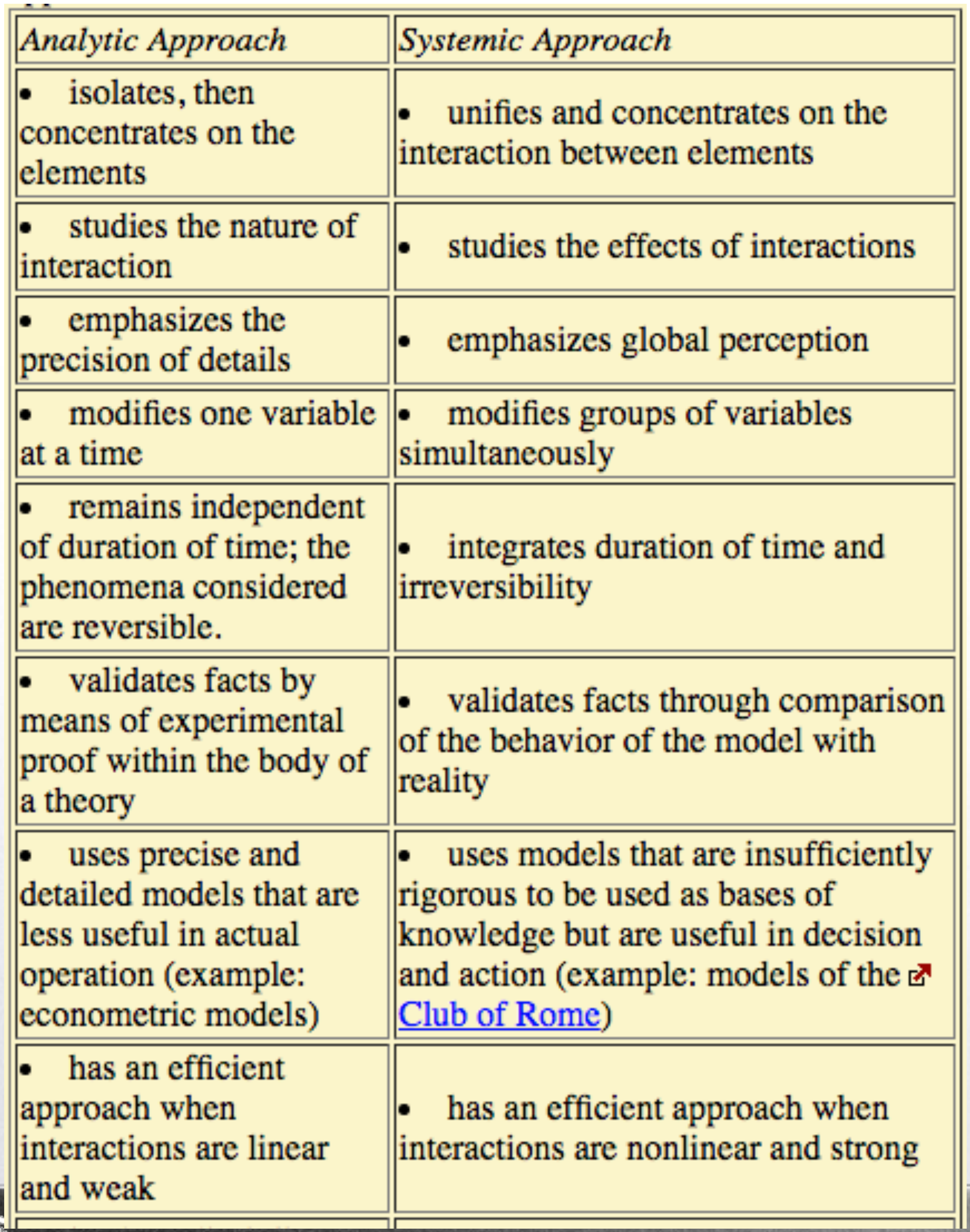
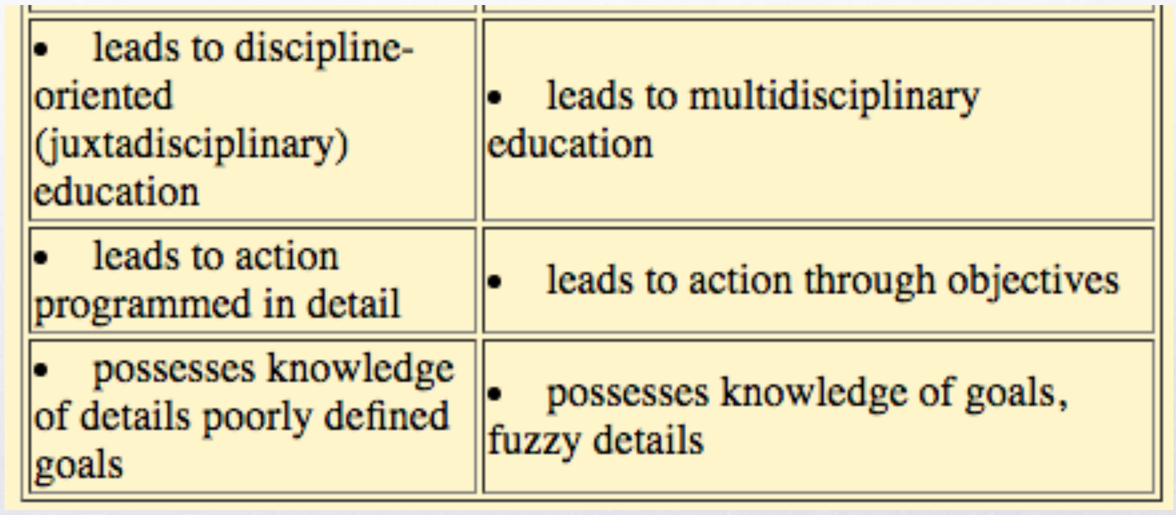

Yoshikawa, H. (1981). General Design Theory and a CAD system, In: *Man-Machine Communication in CAD/CAM*, Sata, T. and Warman, E. (eds.), pp. 35–58, North-Holland, Amsterdam.


Axiom 1 (Axiom of Recognition) Any entity can be recognized or described by the attributes.

Axiom 2 (Axiom of Correspondence) The entity set S' and the set of concept of entity (ideal) S have one-to-one correspondence.

Axiom 3 (Axiom of Operation) The set of abstract concept is a topology of the set of entity concept.

Alguns aspectos práticos

Analytic Approach	*Systemic Approach*
• isolates, then concentrates on the elements	• unifies and concentrates on the interaction between elements
• studies the nature of interaction	• studies the effects of interactions
• emphasizes the precision of details	• emphasizes global perception
• modifies one variable at a time	• modifies groups of variables simultaneously
• remains independent of duration of time; the phenomena considered are reversible.	• integrates duration of time and irreversibility
• validates facts by means of experimental proof within the body of a theory	• validates facts through comparison of the behavior of the model with reality
• uses precise and detailed models that are less useful in actual operation (example: econometric models)	• uses models that are insufficiently rigorous to be used as bases of knowledge but are useful in decision and action (example: models of the Club of Rome)
• has an efficient approach when interactions are linear and weak	• has an efficient approach when interactions are nonlinear and strong
• leads to discipline-oriented (juxtadisciplinary) education	• leads to multidisciplinary education
• leads to action programmed in detail	• leads to action through objectives
• possesses knowledge of details poorly defined goals	• possesses knowledge of goals, fuzzy details

Portanto, um bom projeto de sistema começa com uma descrição geral de objetivos que pode ser colocada em um WBS ou em qualquer outro formato. Este seria o "modelo (abstrato) original".

Mas, na maior parte dos casos um novo sistema é requerido para “substituir" um outro já existente que se tornou obsoleto. Portanto o processo deve começar pela modelagem do “system-as-is”, isto é, do sistema legado. Os “novos requisitos” devem então demarcar as diferenças entre os “system-as-is” e o “system-to-be”.

E o que acontece se não existir o "system-as-is"? Existiria um processo de inovação geral, onde um novo sistema é demandado pela primeira vez?

Se não existir o "system-as-is", então faça um modelo com os requisitos extraídos do stakeholder. Este será o system-as-is! O system-to-be será o resultado da otimização e engenharia das soluções após alguma eventual negociação.

Fundamentação matemática para o MBSE

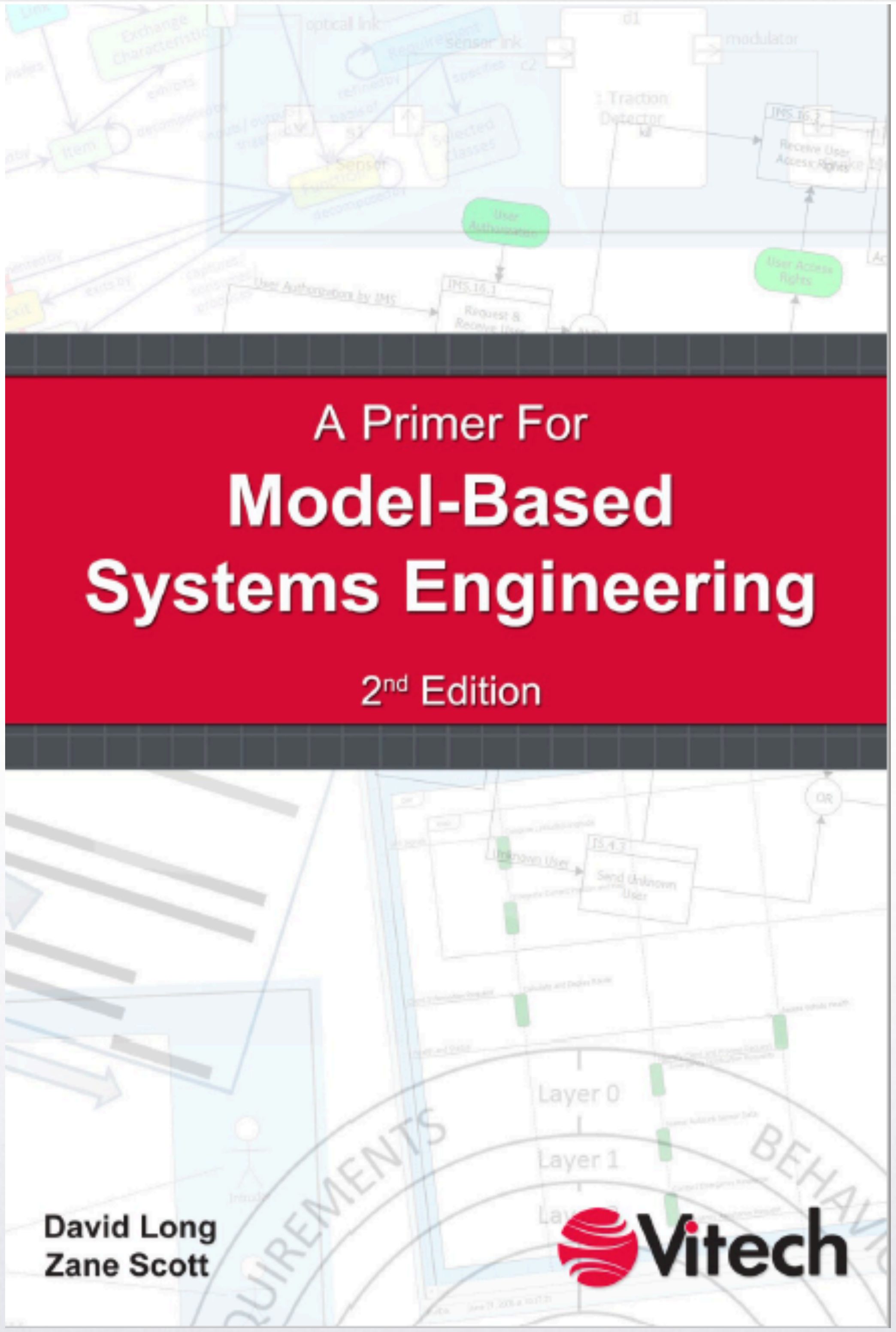
A. Wayne Waymore (T3SD) Tricotyledon Theory of System Design

Wymore, A. Wayne, A Mathematical Theory of Systems Engineering: The Elements , John Wiley & Sons: New York, NY, 1967.

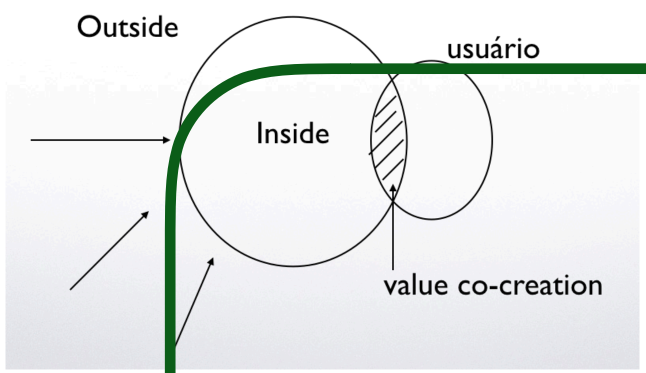
Wymore, A. Wayne, Model-Based Systems Engineering , CRC Press, Inc.: Boca Raton, FL, 1993.

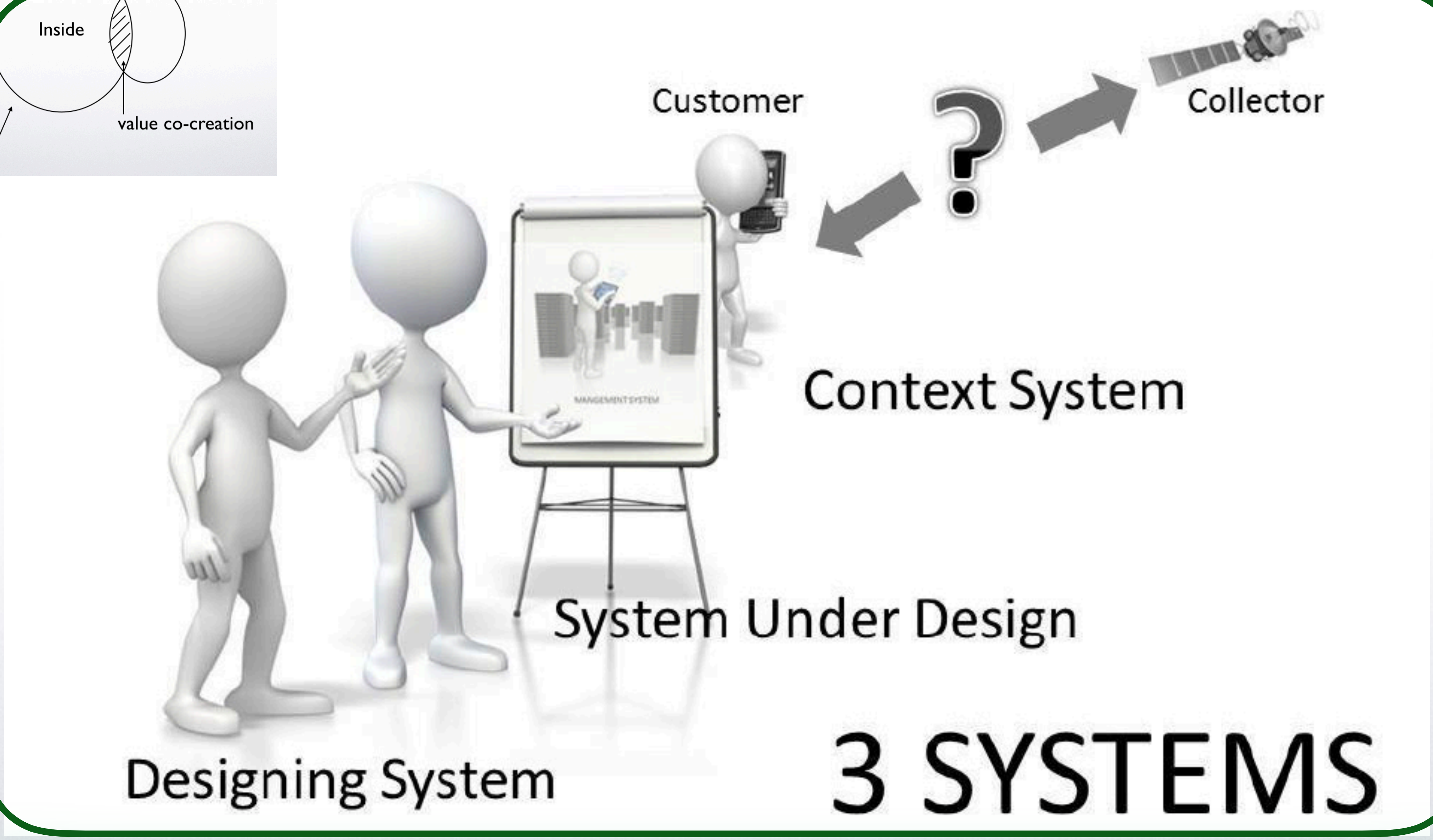
Wymore, A. Wayne, "Contributions to the Mathematical Foundations of Systems Science and Systems Engineering," Systems Movement: Autobiographical Retrospectives, The University of Arizona, Tucson, AZ, 2004.

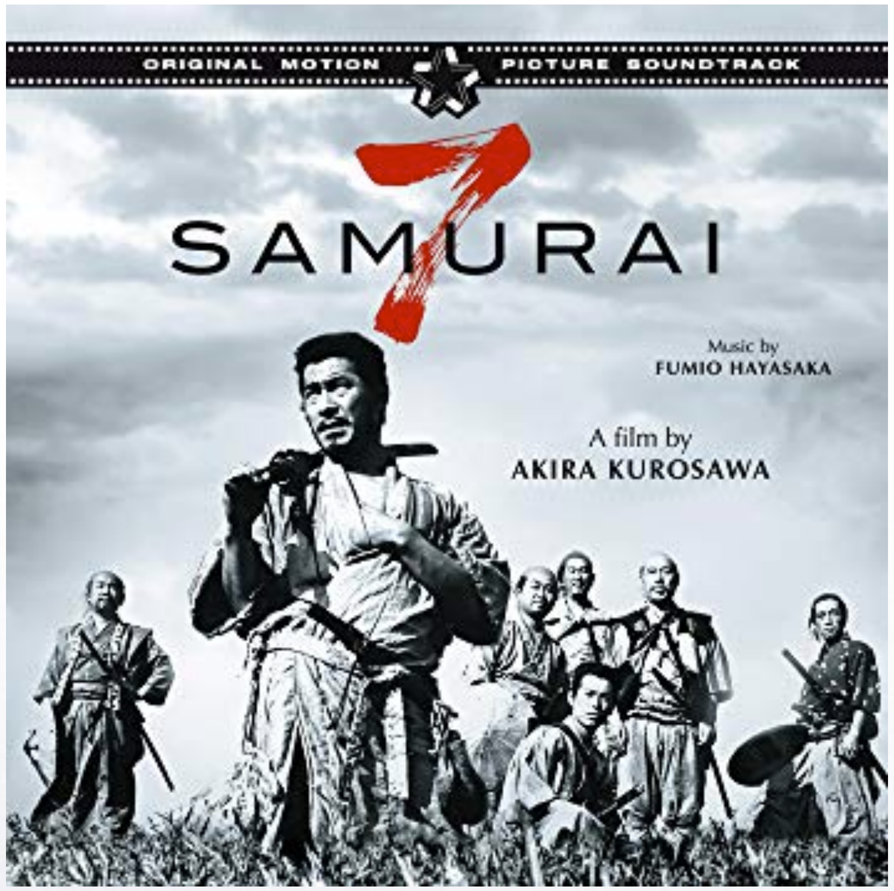
… but, what about the practical approaches?

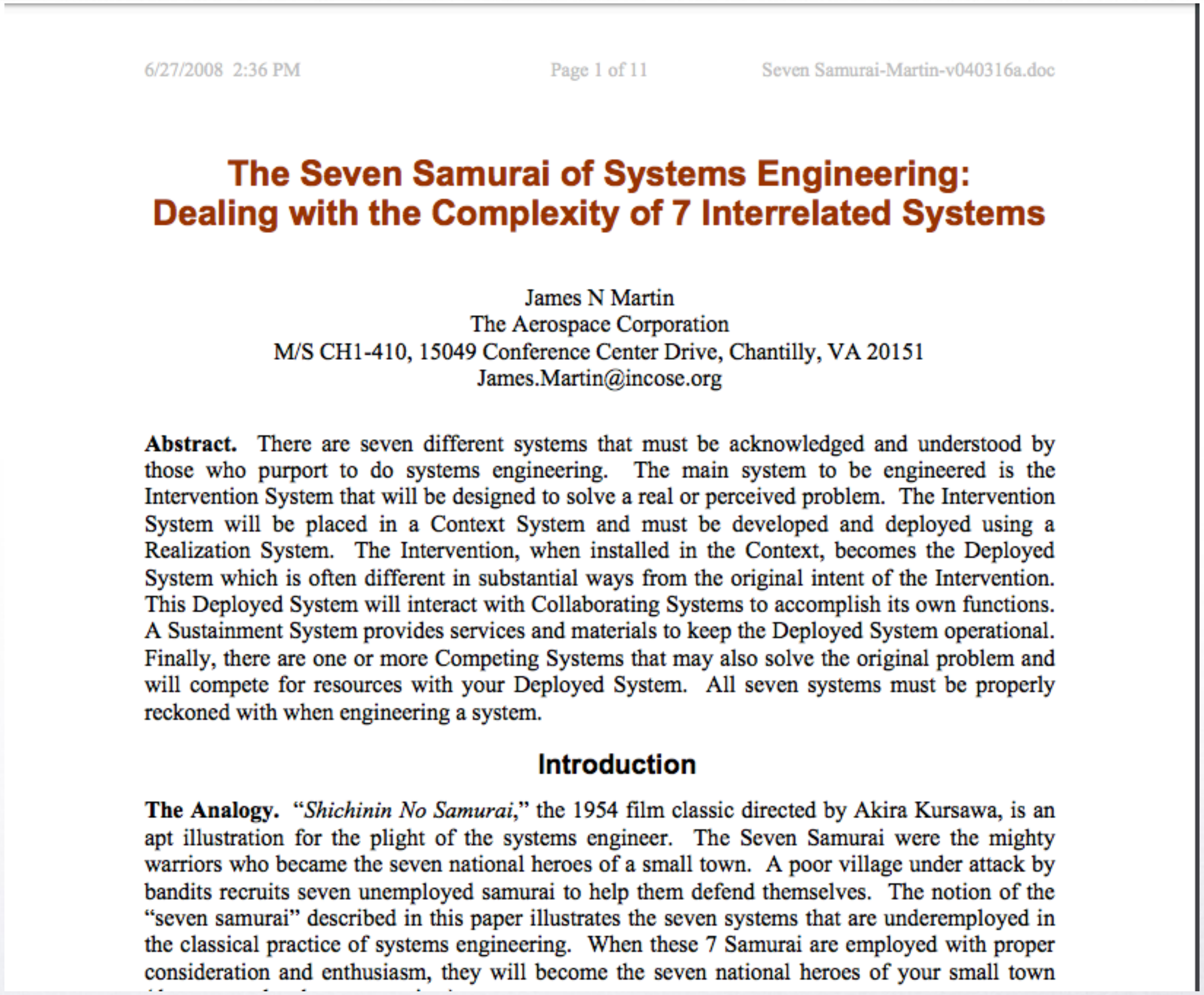

A Primer For
Model-Based
Systems Engineering
2nd Edition
David Long
Zane Scott
Vitech

Development system








6/27/2008 2:36 PM Page 1 of 11 Seven Samurai-Martin-v040316a.doc

The Seven Samurai of Systems Engineering: Dealing with the Complexity of 7 Interrelated Systems

James N Martin
The Aerospace Corporation
M/S CH1-410, 15049 Conference Center Drive, Chantilly, VA 20151
James.Martin@incose.org

Abstract. There are seven different systems that must be acknowledged and understood by those who purport to do systems engineering. The main system to be engineered is the Intervention System that will be designed to solve a real or perceived problem. The Intervention System will be placed in a Context System and must be developed and deployed using a Realization System. The Intervention, when installed in the Context, becomes the Deployed System which is often different in substantial ways from the original intent of the Intervention. This Deployed System will interact with Collaborating Systems to accomplish its own functions. A Sustainment System provides services and materials to keep the Deployed System operational. Finally, there are one or more Competing Systems that may also solve the original problem and will compete for resources with your Deployed System. All seven systems must be properly reckoned with when engineering a system.

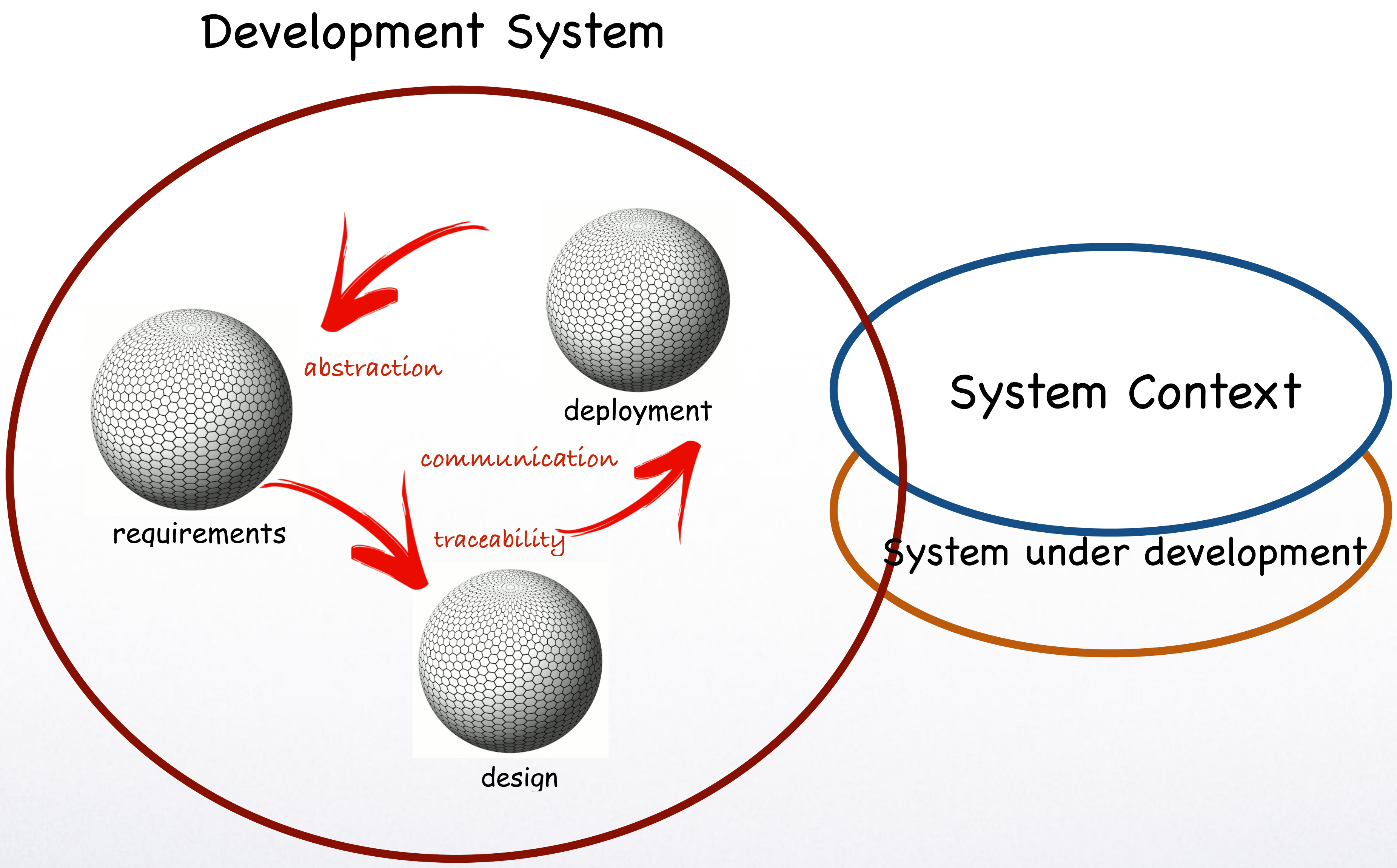
Introduction

The Analogy. "*Shichinin No Samurai*," the 1954 film classic directed by Akira Kursawa, is an apt illustration for the plight of the systems engineer. The Seven Samurai were the mighty warriors who became the seven national heroes of a small town. A poor village under attack by bandits recruits seven unemployed samurai to help them defend themselves. The notion of the "seven samurai" described in this paper illustrates the seven systems that are underemployed in the classical practice of systems engineering. When these 7 Samurai are employed with proper consideration and enthusiasm, they will become the seven national heroes of your small town

Presented at the 2004 Symposium of the International Council on Systems Engineering (INCOSE)
Received "Best Paper" Award

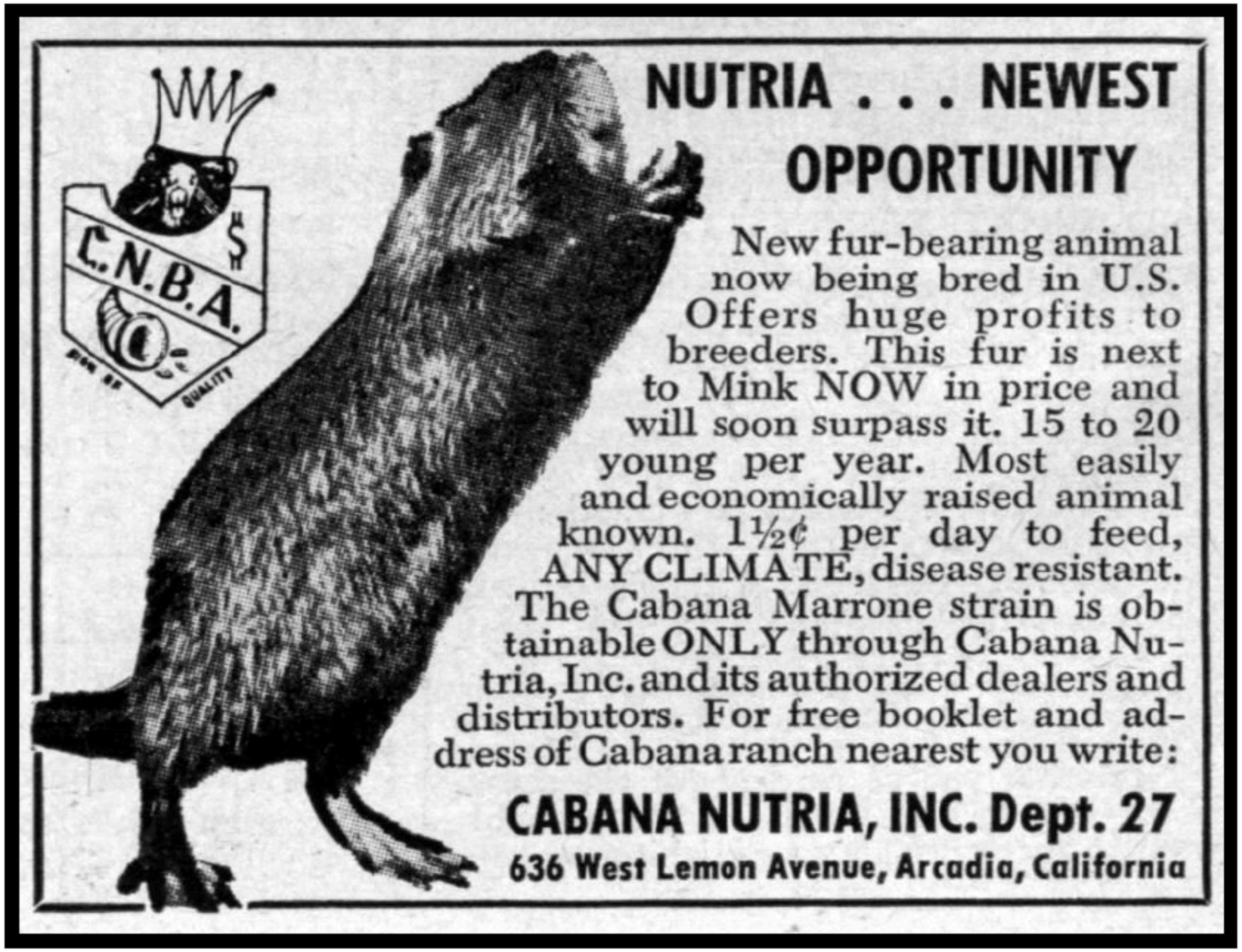
Martin' s systems are:

Context System
Intervention System
Realization System
Deployed System
Collaboration System
Sustainment System
Competition System



A Service view of systems design

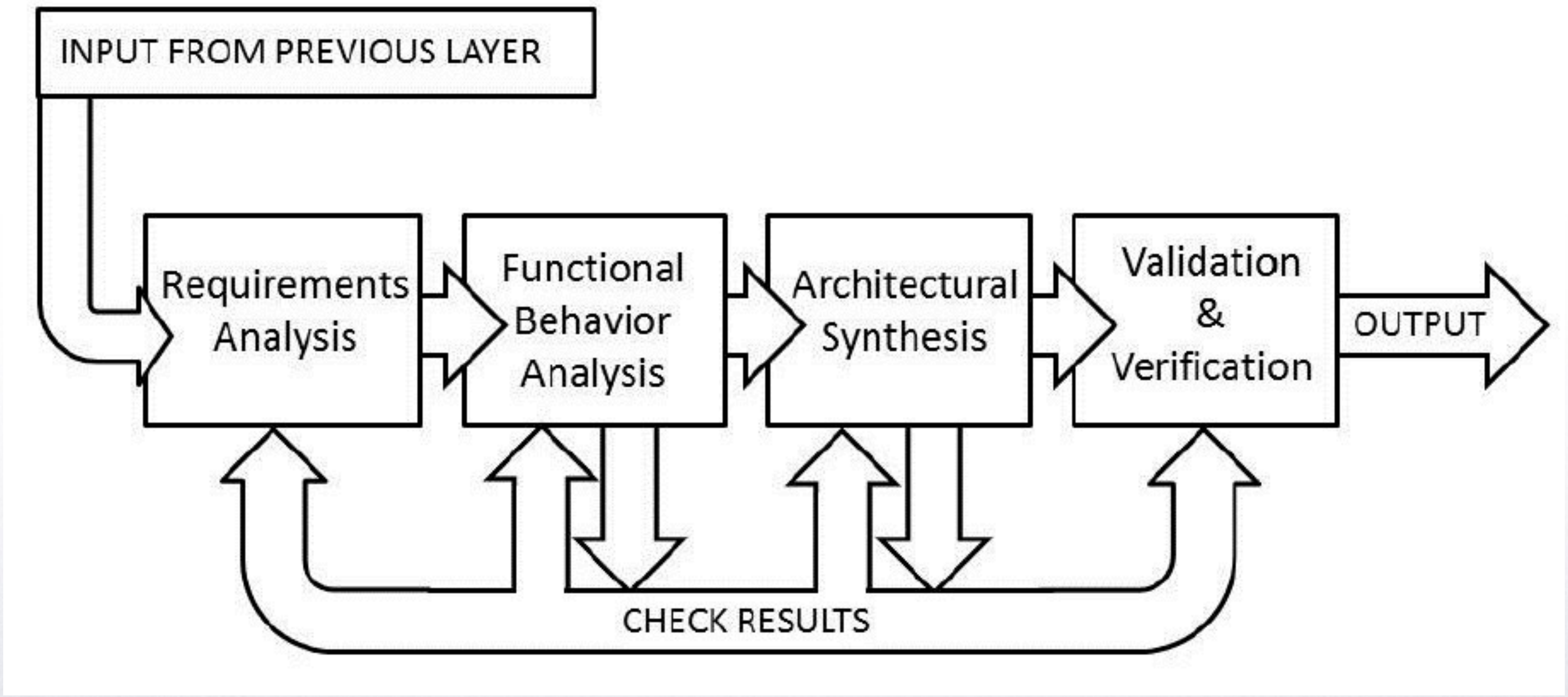
A Primer for Model-Based Systems Engineering


C.N.B.A.
$
SIGN OF QUALITY

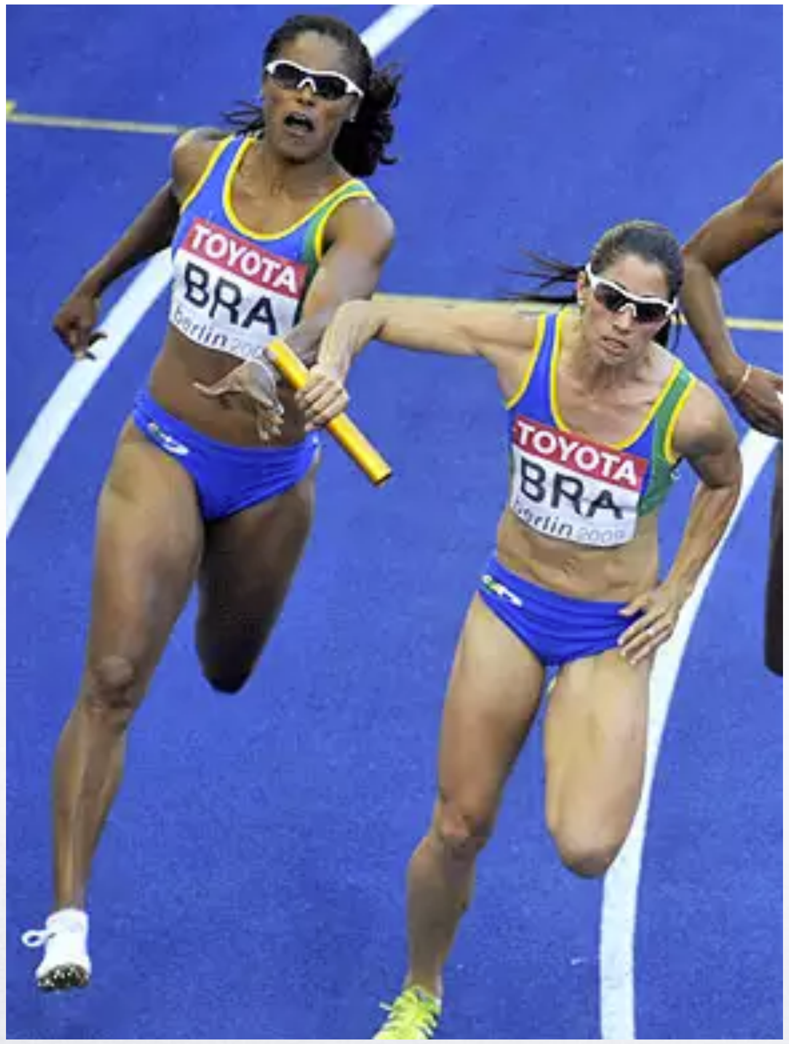
NUTRIA . . . NEWEST OPPORTUNITY

New fur-bearing animal now being bred in U.S. Offers huge profits to breeders. This fur is next to Mink NOW in price and will soon surpass it. 15 to 20 young per year. Most easily and economically raised animal known. 1½¢ per day to feed, ANY CLIMATE, disease resistant. The Cabana Marrone strain is obtainable ONLY through Cabana Nutria, Inc. and its authorized dealers and distributors. For free booklet and address of Cabana ranch nearest you write:

CABANA NUTRIA, INC. Dept. 27
636 West Lemon Avenue, Arcadia, California


INPUT FROM PREVIOUS LAYER
Requirements Analysis
Functional Behavior Analysis
Architectural Synthesis
Validation & Verification
OUTPUT
CHECK RESULTS

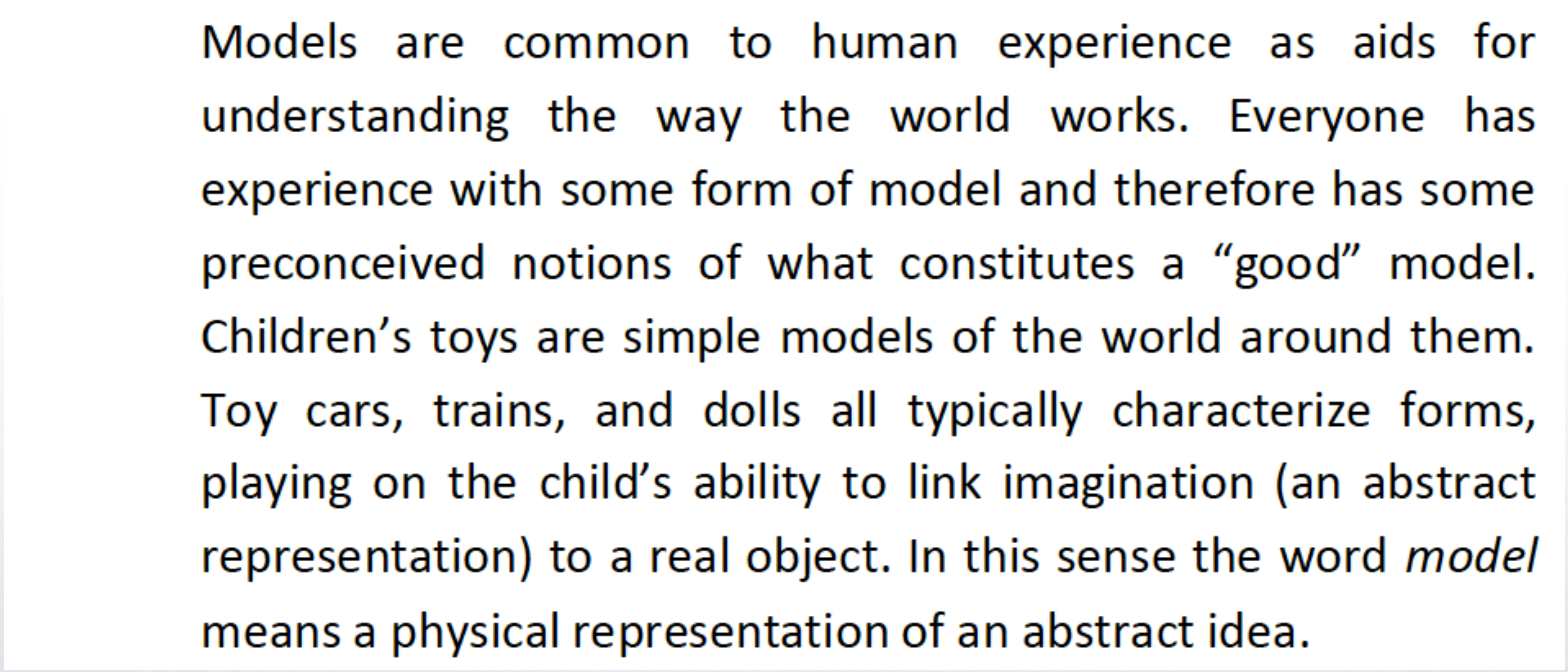
Design Process is a metaphor of a...



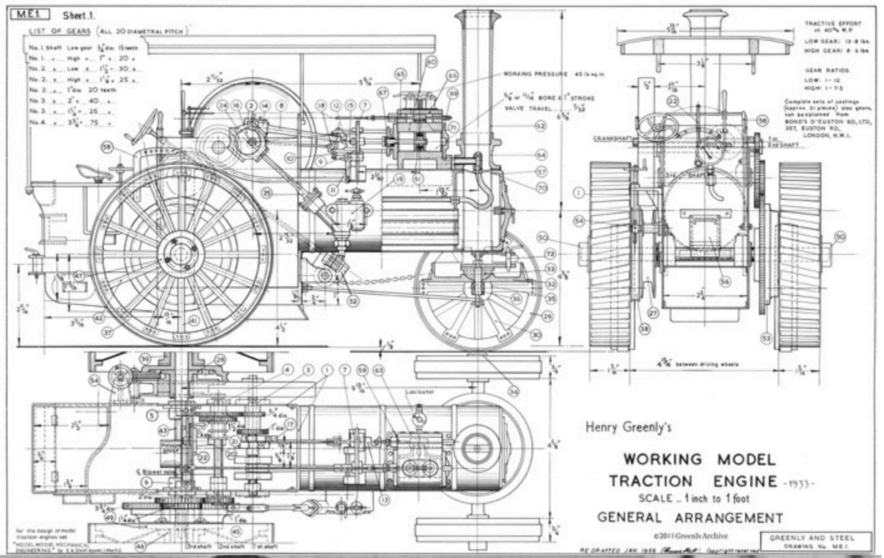
partially ordered relay run...

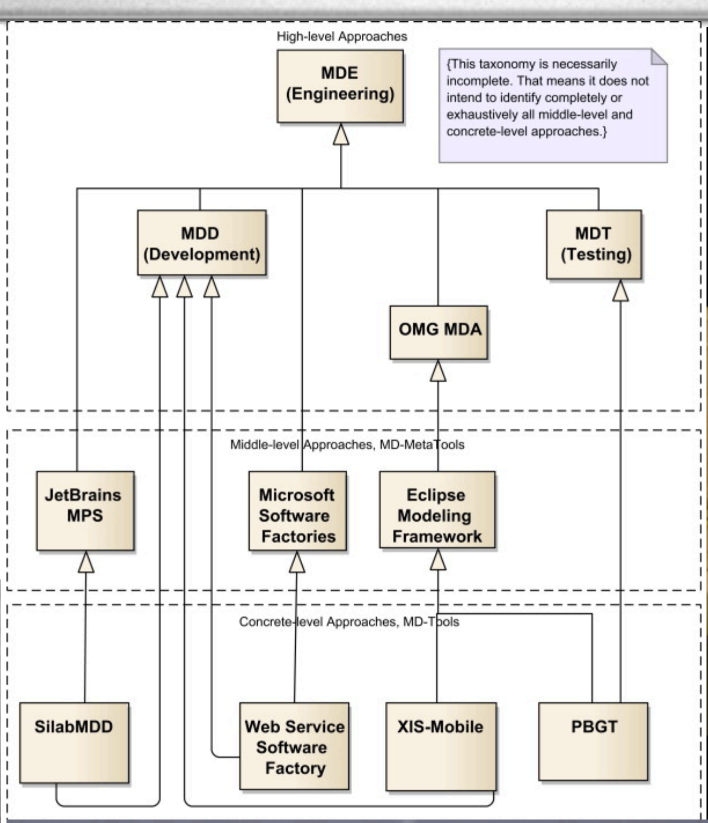
where runners are domains and the stick is a current model.

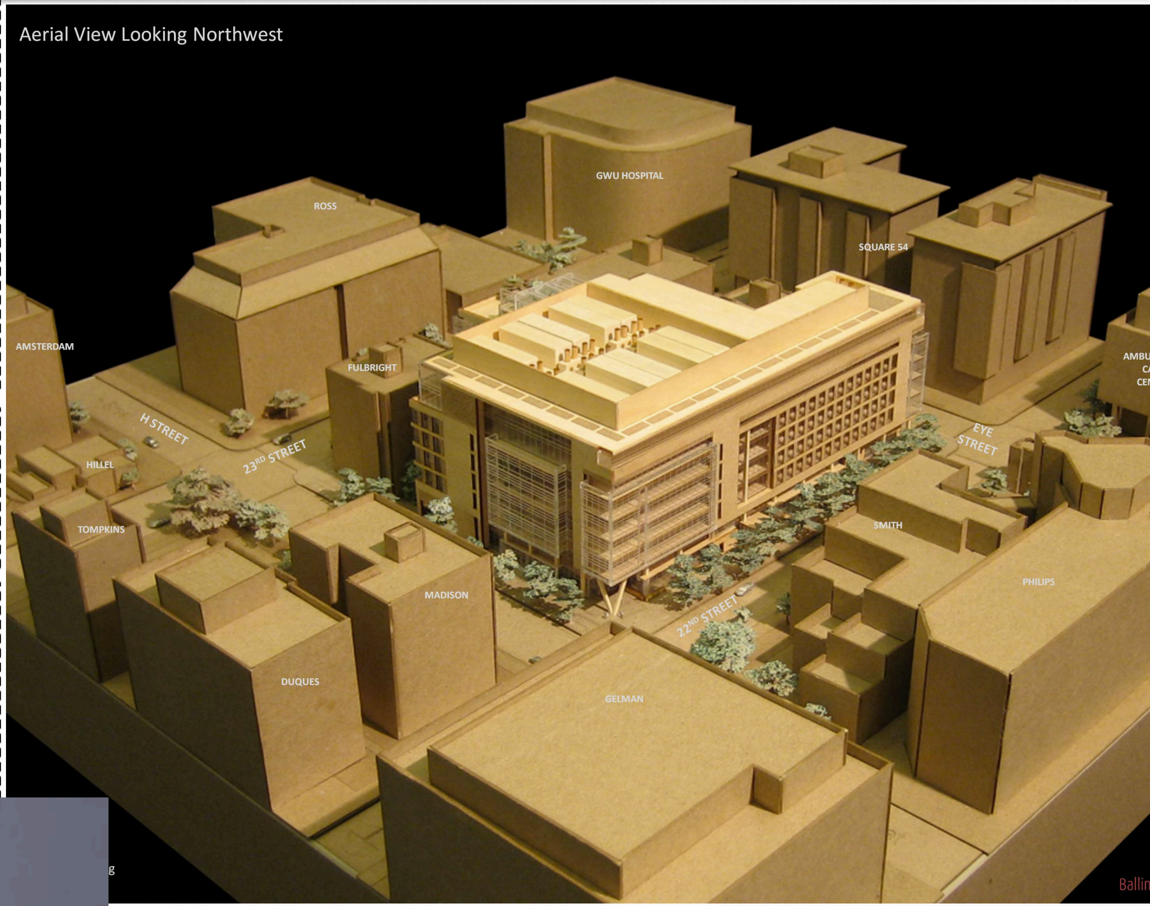
WHAT IS A MODEL?

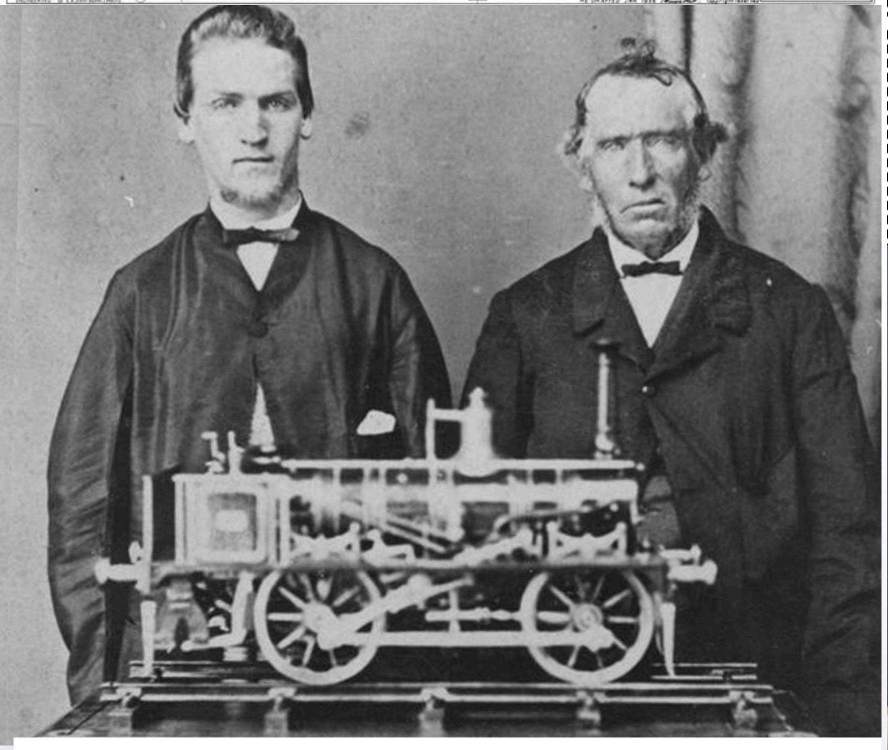


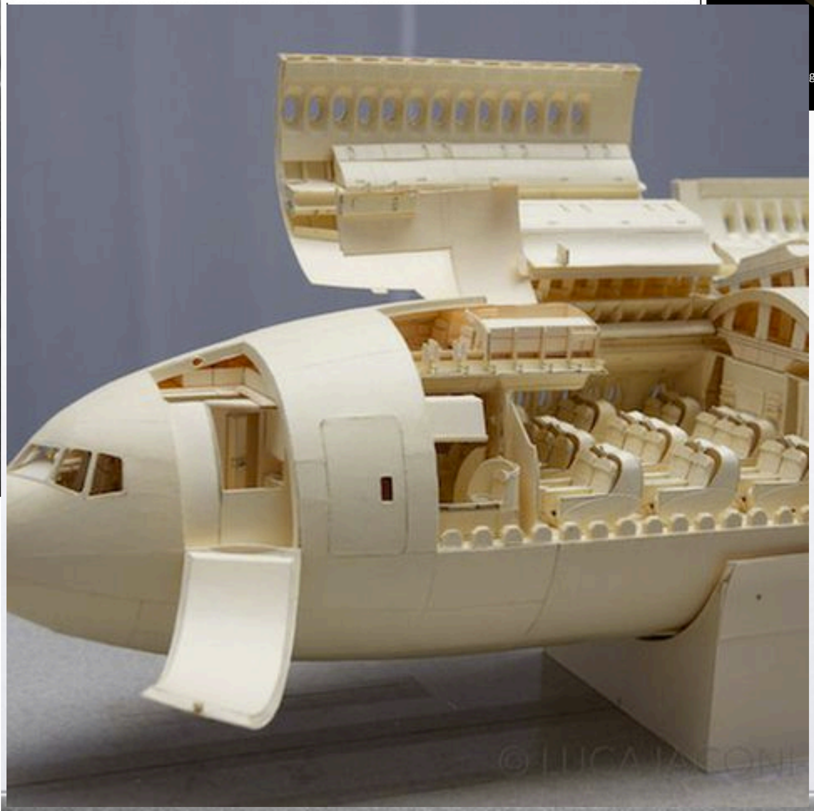
Models are common to human experience as aids for understanding the way the world works. Everyone has experience with some form of model and therefore has some preconceived notions of what constitutes a “good” model. Children’s toys are simple models of the world around them. Toy cars, trains, and dolls all typically characterize forms, playing on the child’s ability to link imagination (an abstract representation) to a real object. In this sense the word *model* means a physical representation of an abstract idea.

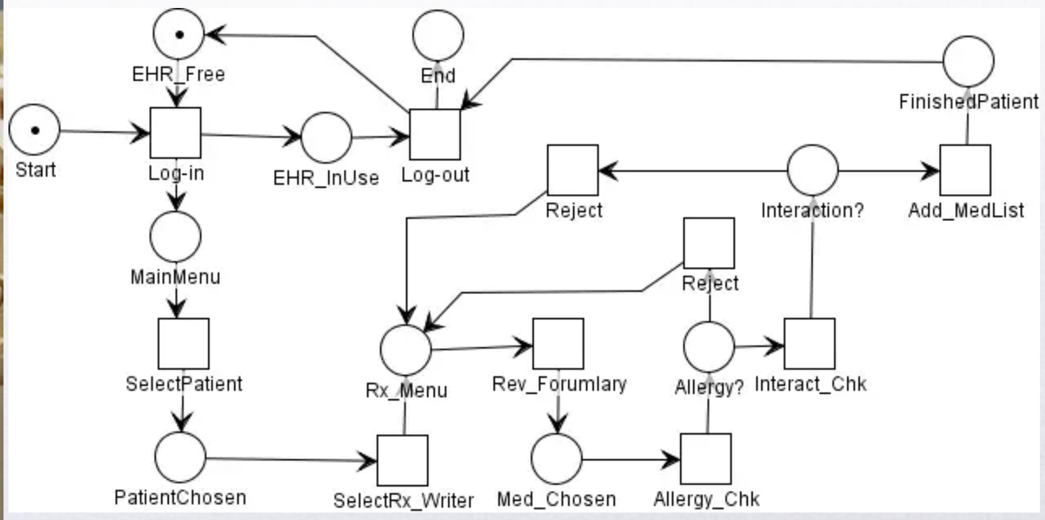



High-level Approaches
MDE (Engineering)
{This taxonomy is necessarily incomplete. That means it does not intend to identify completely or exhaustively all middle-level and concrete-level approaches.}
MDD (Development)
MDT (Testing)
OMG MDA
Middle-level Approaches, MD-MetaTools
JetBrains MPS
Microsoft Software Factories
Eclipse Modeling Framework
Concrete-level Approaches, MD-Tools
SilabMDD
Web Service Software Factory
XIS-Mobile
PBGT









$$(1)\quad \frac{d[LuxI]}{dt} = \alpha_{pTet} \cdot \left(\delta_{pTet} + \left(\frac{1-\delta_{pTet}}{1+\left(\frac{k_{pTet}}{aTc}\right)^{\eta_{pTet}}}\right)\right) - \left(\gamma_{LuxI} + \mu \cdot \left(\frac{N_{max}-N}{N_{max}}\right)\right) \cdot LuxI;$$

$$(2)\quad \frac{d[AiiA]}{dt} = \alpha_{pLux} \cdot \left(\delta_{pLux} + \left(\frac{1-\delta_{pLux}}{1+\left(\frac{k_{pLux}}{HSL}\right)^{\eta_{pLux}}}\right)\right) - \left(\gamma_{AiiA} + \mu \cdot (\frac{N_{max}-N}{N_{max}})\right) \cdot AiiA;$$

$$(3)\quad \frac{d[HSL]}{dt} = N \cdot V_{max} \cdot \frac{1}{1+\frac{k_{M,LuxI}}{LuxI}} - N \cdot \frac{k_{cat} \cdot AiiA \cdot HSL}{k_{M,AiiA} + AiiA} - \gamma_{HSL} \cdot HSL;$$

$$(4)\quad \frac{d[N]}{dt} = \mu \cdot N \cdot \left(\frac{N_{max}-N}{N_{max}}\right);$$

Model Structure

Language (SDL)

Structure

Argumentation

Presentation

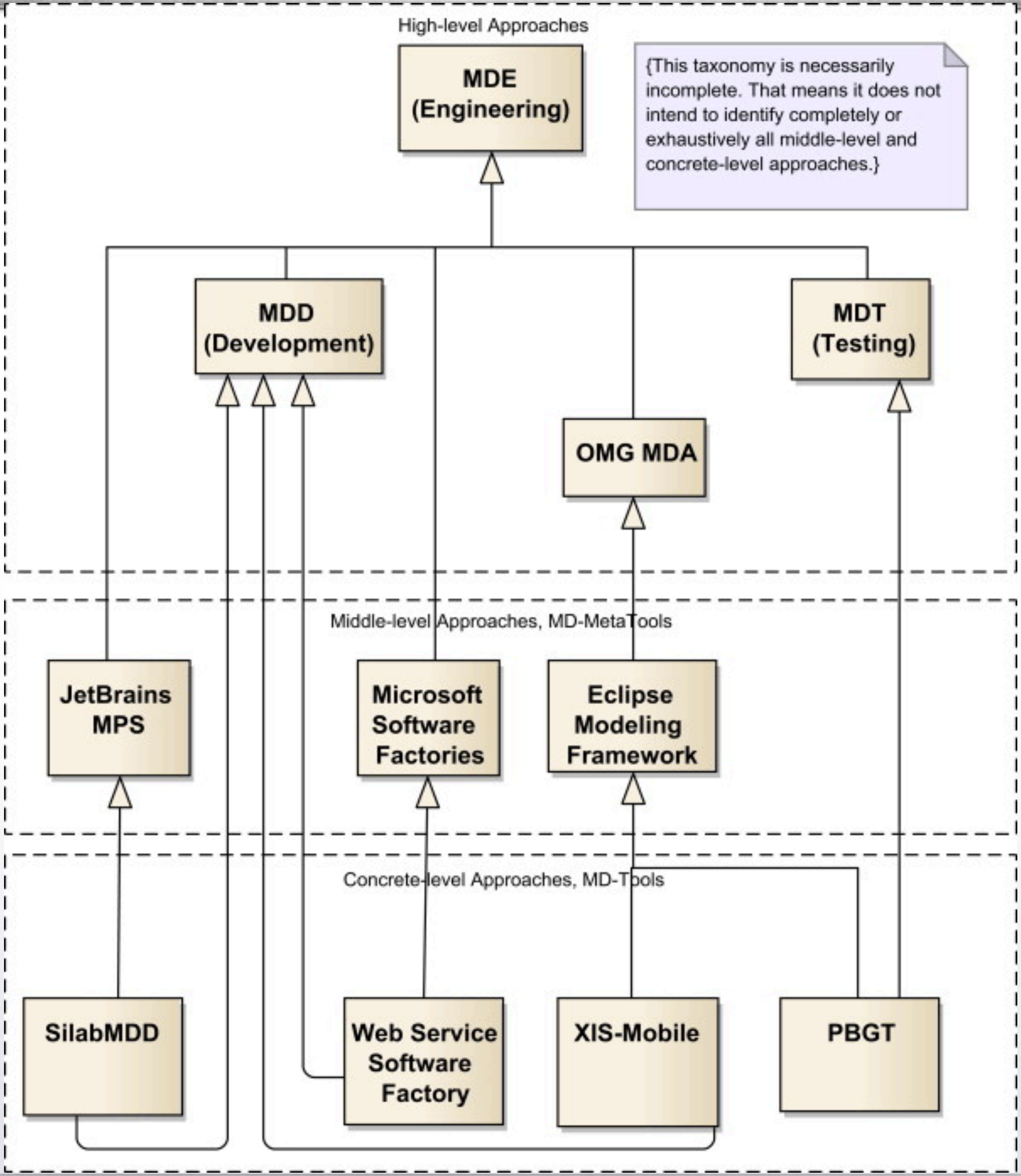
Model Expectations

Ordering

Persuasive Power

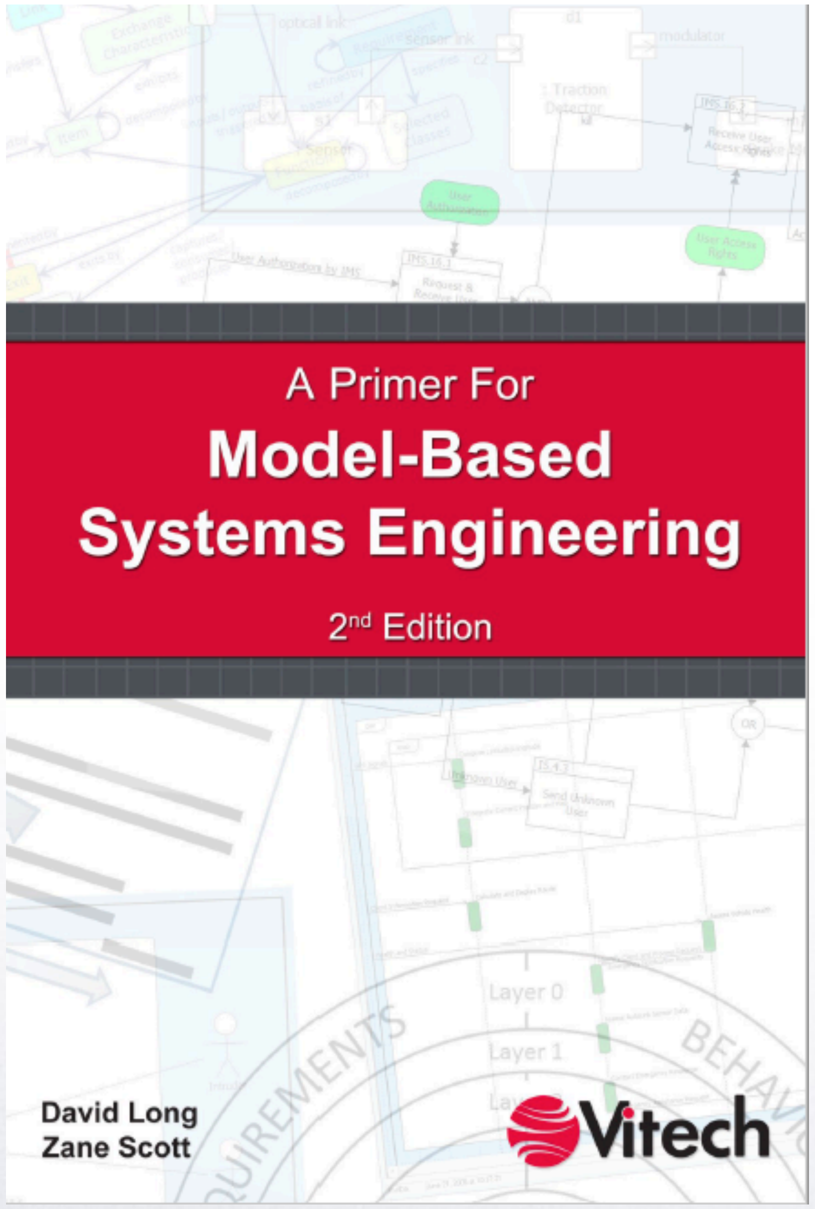
Consistency

Prediction Power


High-level Approaches
MDE (Engineering)
{This taxonomy is necessarily incomplete. That means it does not intend to identify completely or exhaustively all middle-level and concrete-level approaches.}
MDD (Development)
MDT (Testing)
OMG MDA
Middle-level Approaches, MD-MetaTools
JetBrains MPS
Microsoft Software Factories
Eclipse Modeling Framework
Concrete-level Approaches, MD-Tools
SilabMDD
Web Service Software Factory
XIS-Mobile
PBGT

The big challenge in today's class is to answer the question: how could be scale the concepts discussed so far to large and complex (automated) systems, indeed, how could we apply all this knowledge - claimed to be effective - to real life projects?

→ Model Driven Engineering



Founder and President of Vitech Co. since 1992. Received an Engineering Bachelor from Virginia Tech and a Master from the same University. Member of the Board of Directors of INCOSE since 2003.

Vice-President of Professional Services in Vitech Co. since 2009. Received a Bachelor fin Economics from Virginia Tech and a Doctor in Law from University of Tenessee. Member of the Corporate Advisory Board of INCOSE.

Obrigado

Reinaldo