PMR 5020

Metodologia do Projeto de Sistemas

Aula 5: Métodos modernos de modelagem de requisitos

Prof. José Reinaldo Silva
reinaldo@usp.br

A fase de requisitos e a ER

Como vimos até aqui:

i) a fase de requisitos é muito importante para os projetos;
ii) o sucesso do projeto depende de uma boa ER;
iii) em especial para sistemas esta fase é primordial;
iv) esta fase é composta da : eliciação, modelagem e análise de requisitos;

The key problems

- The priority of requirements from different viewpoints changes during the development process.
- System customers may specify requirements from a business perspective that conflict with end-user requirements.
- The business and technical environment of the system changes during its development.

Good req's X Bad req's

- Enduring requirements. Stable requirements derived from the core activity of the customer organisation. E.g. a hospital will always have doctors, nurses, etc. May be derived from domain models
- Volatile requirements. Requirements which change during development or when the system is in use. In a hospital, requirements derived from health-care policy

The traceability concept

- Traceability is concerned with the relationships between requirements, their sources and the system design
- Source traceability
  - Links from requirements to stakeholders who proposed these requirements;
- Requirements traceability
  - Links between dependent requirements;
- Design traceability
  - Links from the requirements to the design;

Traceability

"The requirements traceability is the ability to describe and follow the life of a requirement, in both a forward and backward direction, i.e. from its origins, through its development and specification, to its subsequent deployment and use, and through periods of ongoing refinement and iteration in any of these phases."

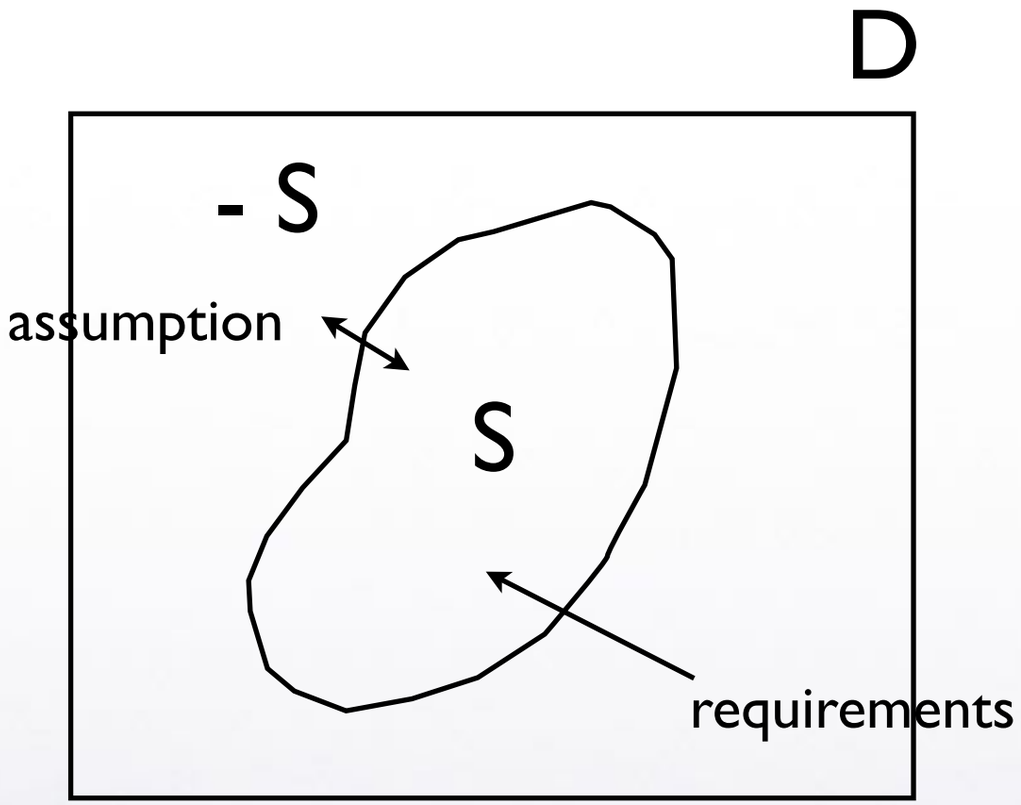
Gotel, O., Filkenstein, A.; An Analysis of the Requirements Traceability Problem, in Proc. of the First Int. Conf. on Requirements Engineering, pp 94-101, Colorado Springs, USA, 1994.

Design features

A base para a formulação de requisitos deve ser um conjunto genérico de features (objetivos) abstratos que definem o projeto. Esta definição genérica pode ser refinada em requisitos funcionais e não-funcionais até chegar em funções específicas de mais baixo nível.

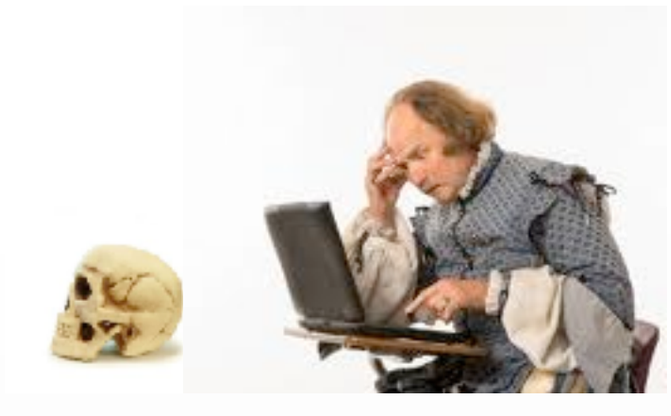
Silva, J.R.; Uma formalização do processo de design baseado em metáforas: sua aplicação na automatização de Sistemas de Eventos Discretos, tese de doutorado, Escola Politécnica da USP, 1992.

Contexto

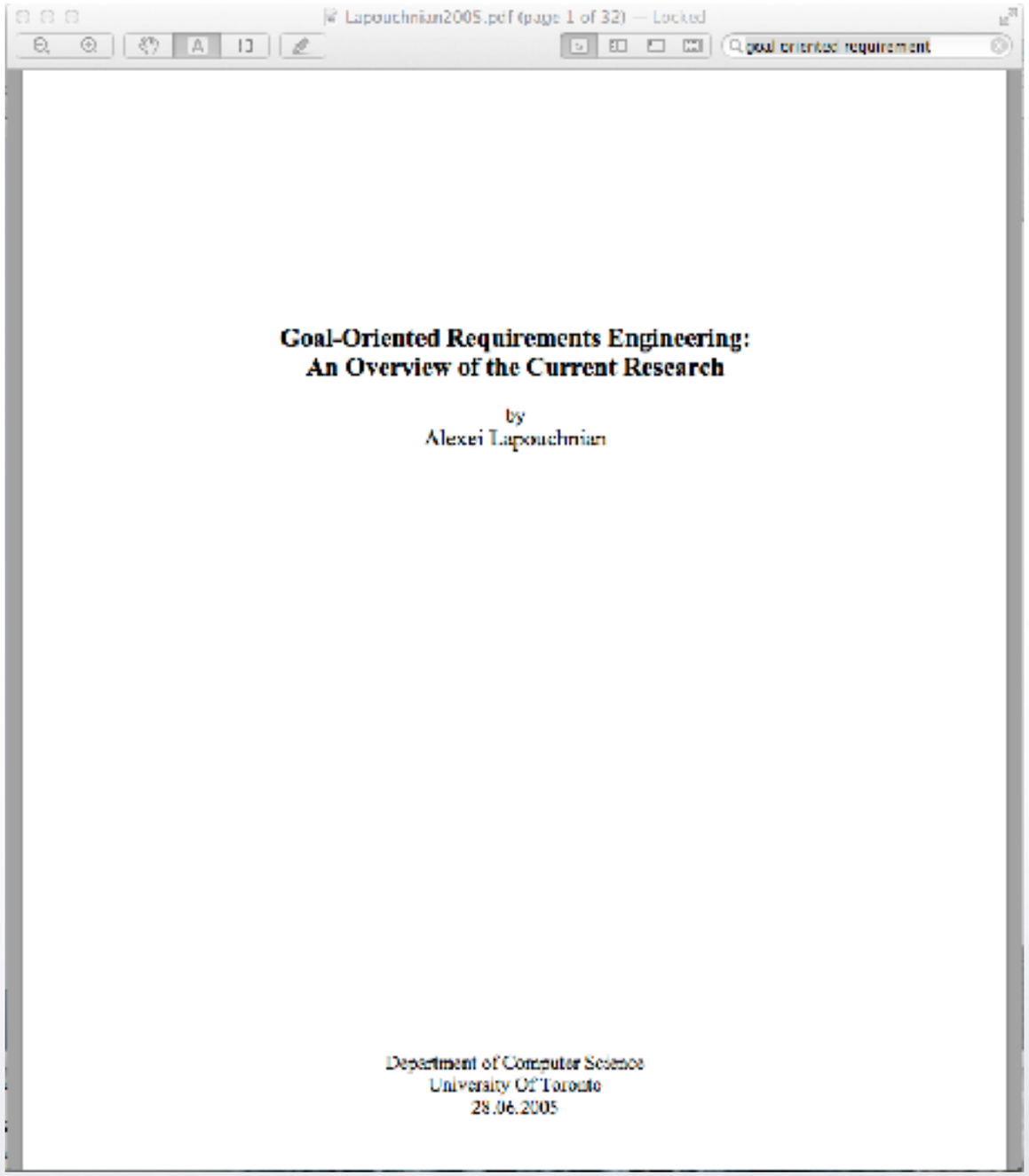


Dominio de aplicação

Leitura da semana



To be funcional or not to be funcional, that is the question


Lapouchnian2005.pdf (page 1 of 32) — Locked

goal oriented requirement

Goal-Oriented Requirements Engineering: An Overview of the Current Research

by
Alexei Lapouchnian

Department of Computer Science
University Of Toronto
28.06.2005

Os perigos da eliciação de requisitos

funcional

não-funcional

Um dos perigos da eliciação (e portanto da análise) de requisitos é privilegiar a análise funcional e a funcionalidade como alvo central.
Normalmente se diz que os requisitos devem refletir o que o sistema é e o que faz mas NUNCA "o como" este deve fazer as coisas (que seria alvo de etapas posteriores).
O perigo do "funcional" como antecipação do design está no stakeholder (especialmente em domínios disjuntos) ou no próprio designer (no caso de domínios próximos), ou em ambos quando não se trata de casos extremos.

Requisitos orientados a objetivos (goals)

What are goals?
A *goal* is an objective the system under consideration should achieve. Goal formulations thus refer to intended properties to be ensured; they are optative statements as opposed to indicative ones, and bounded by the subject matter

Axel van Lamsweerde, Goal Oriented Requirement Engineering: a Guided Tour, RE 2001, Toronto

Exemplos

Atender mais passageiros

Agilizar e personalizar o atendimento

garantir a circulação com segurança e rapidez

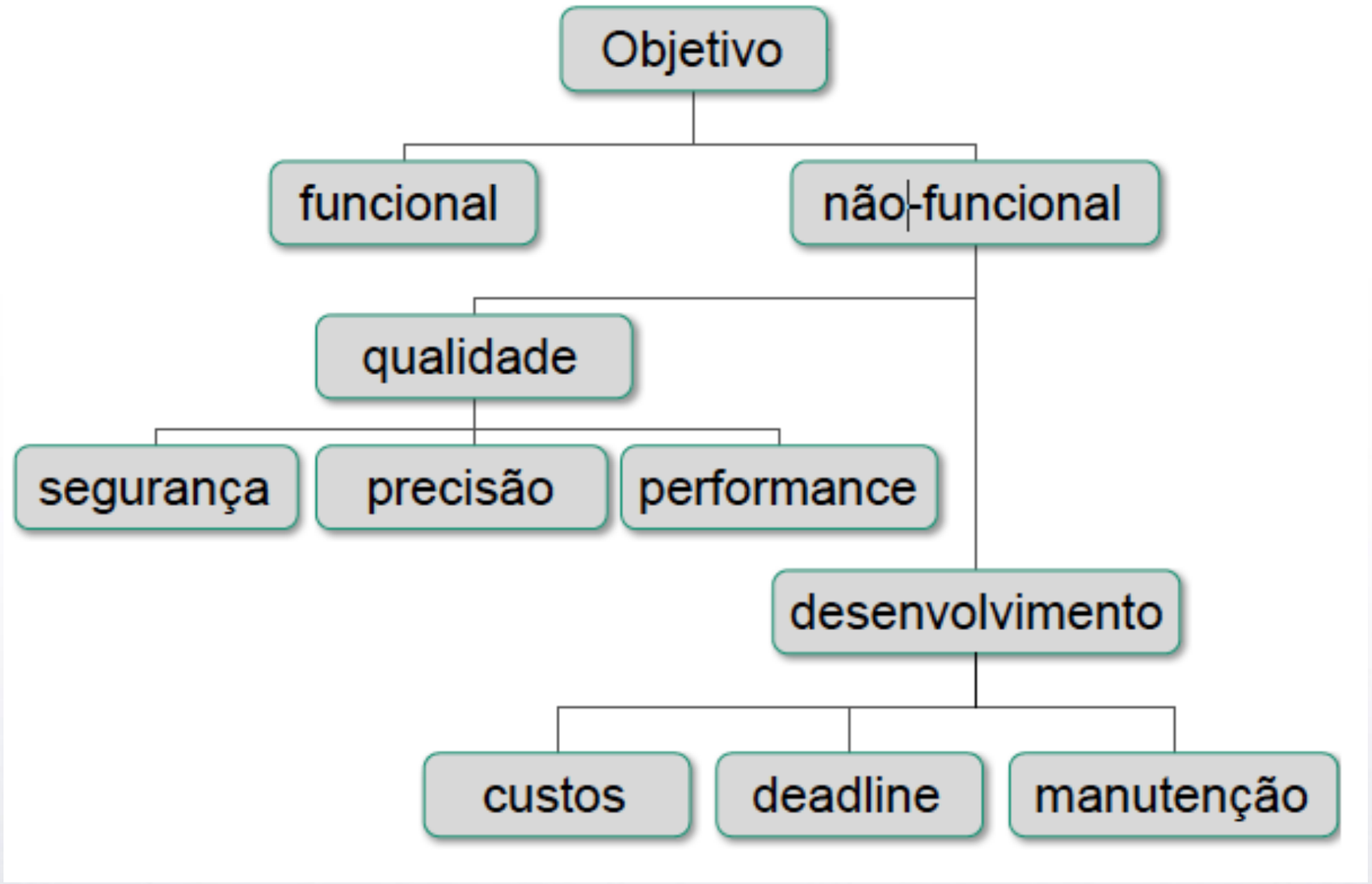
Goals (objetivos) integram o funcional e o não funcional

funcional

não-funcional

Goals also cover different types of concerns: functional concerns associated with the services to be provided, and non- functional concerns associated with quality of service such as safety, security, accuracy, performance, and so forth.

Uma estratégia mais inteligente seria basear a busca pelos requisitos em conceitos que integrem o funcional e o não-funcional evitando o dilema da escolha ou da prioridade.


Objetivo
funcional
não-funcional
qualidade
segurança
precisão
performance
desenvolvimento
custos
deadline
manutenção

Porque goals (objetivos)?

Why are goals needed?
There are many reasons why goals are so important in the RE process.
• Achieving requirements completeness is a major RE concern.. Goals provide a precise criterion for *sufficient completeness* of a requirements specification; the specification is complete with respect to a set of goals if all the goals can be proved to be achieved from the specification and the properties known about the domain considered.

O stakeholder é o "único" que pode encerrar/validar um ciclo de requisitos

É difícil distinguir quando um requisito que se refere a algum aspecto do sistema foi plenamente eliciado. Assim a preocupação inicial que tinhamos com o sistema e sua representação por um plano pode se propagar para todo o processo de eliciação.

• Avoiding irrelevant requirements is another major RE concern. Goals provide a precise criterion for requirements *pertinence*; a requirement is pertinent with respect to a set of goals in the domain considered if its specification is used in the proof of one goal at least.

• Explaining requirements to stakeholders is another important issue. Goals provide the rationale for requirements, in a way similar to design goals in design processes [Mos85, Lee91]. A requirement appears because of some underlying goal which provides a base for it [Ros77, Dar91, Som97]. More explicitly, a goal refinement tree provides traceability links from high-level strategic objectives to low-level technical requirements. In particular, for business application systems, goals may be used to relate the software-to-be to organizational and business contexts

Objetividade é outro aspecto importante. É preciso ter um conjunto minimal de requisitos válidos.

facilita o reuso

Validação e tracebility (discutido na aula passada) são outros aspectos importantes para uma boa fase de eliciação e análise de requisitos.

facilita a manutenção

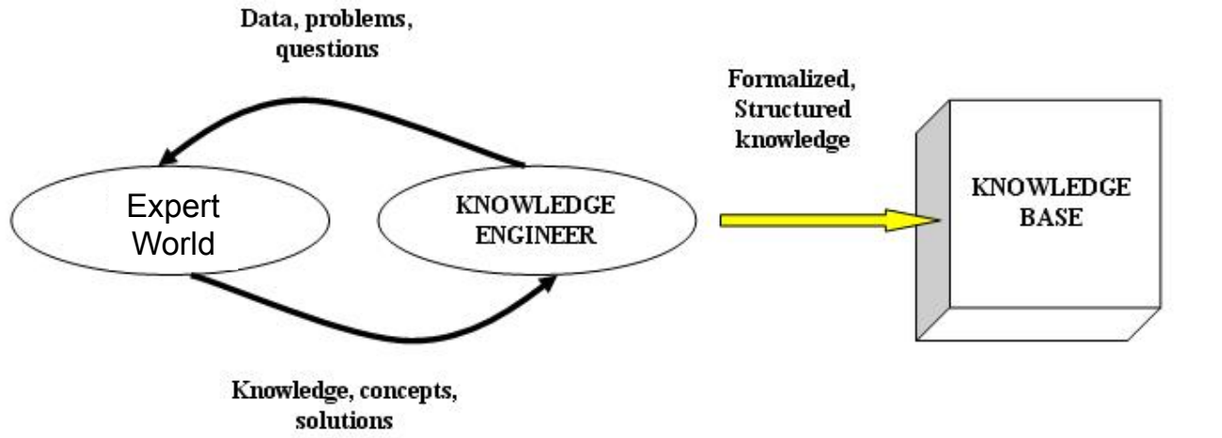
KAOS: Knowledge Acquisition in Automated Specification

KAOS is one method for formalising goals into requirements

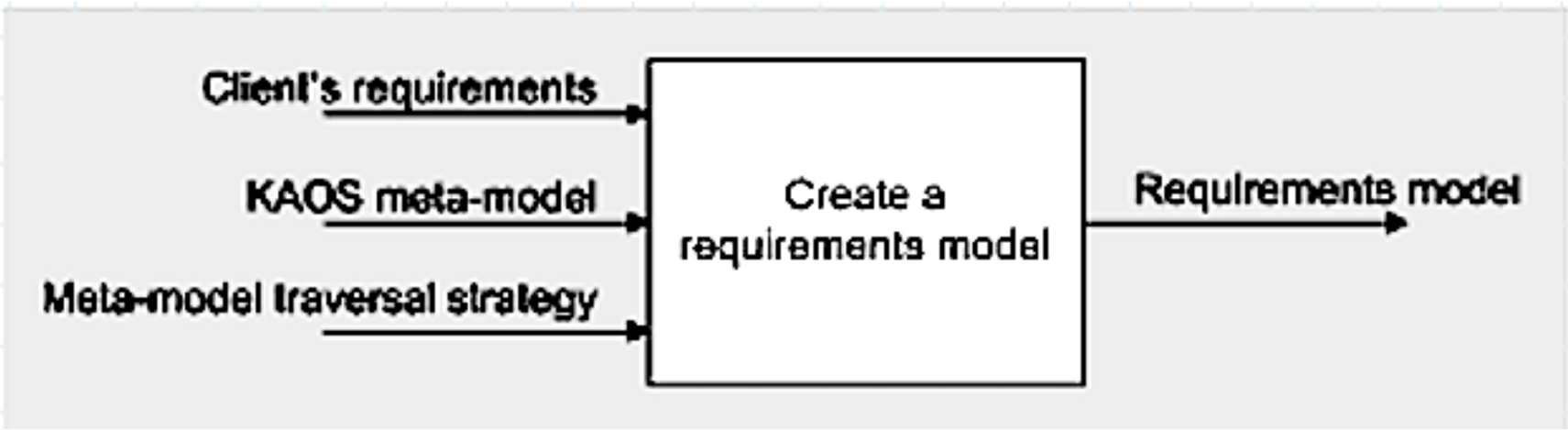
KAOS is a research project, but it is well documented, and has tool support from a tool called GRAIL

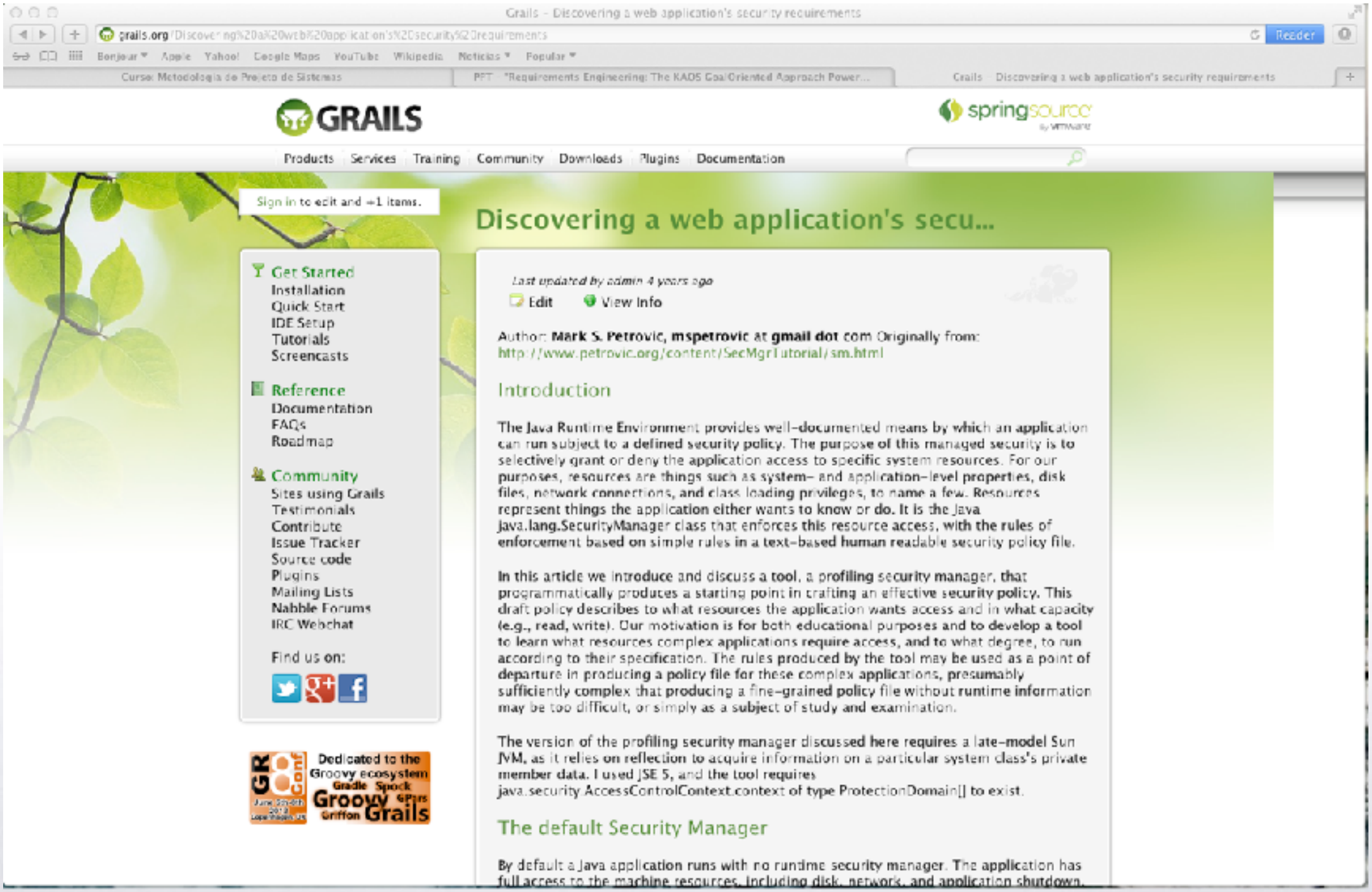
The aim of the ***KAOS approach*** is to derive a description of a system's behaviour and an initial analysis of its structure through acquiring and formalising functional and non-functional goals for a composite system

Stewart Green



KAOS estruturado




Grails - Discovering a web application's security requirements
grails.org/Discovering%20a%20web%20application's%20security%20requirements
Reader
Bonjour
Apple
Yahoo!
Google Maps
YouTube
Wikipedia
Noticias
Popular
Curso: Metodologia de Projeto de Sistemas
PPT "Requirements Engineering: The KAOS GoalOriented Approach Power...
Grails - Discovering a web application's security requirements
GRAILS
springsource
Products
Services
Training
Community
Downloads
Plugins
Documentation
Sign in to edit and +1 items.
Discovering a web application's secu...
Get Started
Installation
Quick Start
IDE Setup
Tutorials
Screencasts
Reference
Documentation
FAQs
Roadmap
Community
Sites using Grails
Testimonials
Contribute
Issue Tracker
Source code
Plugins
Mailing Lists
Nabble Forums
IRC Webchat
Find us on:
Dedicated to the Groovy ecosystem
Gradle Spock
Groovy GPars
Griffon Grails
Last updated by admin 4 years ago
Edit
View Info
Author: Mark S. Petrovic, mspetrovic at gmail dot com Originally from:
http://www.petrovic.org/content/SecMgrTutorial/sm.html
Introduction
The Java Runtime Environment provides well-documented means by which an application can run subject to a defined security policy. The purpose of this managed security is to selectively grant or deny the application access to specific system resources. For our purposes, resources are things such as system- and application-level properties, disk files, network connections, and class loading privileges, to name a few. Resources represent things the application either wants to know or do. It is the Java java.lang.SecurityManager class that enforces this resource access, with the rules of enforcement based on simple rules in a text-based human readable security policy file.
In this article we introduce and discuss a tool, a profiling security manager, that programmatically produces a starting point in crafting an effective security policy. This draft policy describes to what resources the application wants access and in what capacity (e.g., read, write). Our motivation is for both educational purposes and to develop a tool to learn what resources complex applications require access, and to what degree, to run according to their specification. The rules produced by the tool may be used as a point of departure in producing a policy file for these complex applications, presumably sufficiently complex that producing a fine-grained policy file without runtime information may be too difficult, or simply as a subject of study and examination.
The version of the profiling security manager discussed here requires a late-model Sun JVM, as it relies on reflection to acquire information on a particular system class's private member data. I used JSE 5, and the tool requires
java.security.AccessControlContext.context of type ProtectionDomain[] to exist.
The default Security Manager
By default a Java application runs with no runtime security manager. The application has full access to the machine resources, including disk, network, and application shutdown.

2.1 Installation Requirements

Before installing Grails you will need as a minimum a Java Development Kit (JDK) installed version 1.6 or above. Download the appropriate JDK for your operating system, run the installer, and then set up an environment variable called `JAVA_HOME` pointing to the location of this installation. If you're unsure how to do this, we recommend the video installation guides from grailsexample.net:

- Windows
- Linux
- Mac OS X

These will show you how to install Grails too, not just the JDK.

On some platforms (for example OS X) the Java installation is automatically detected. However in many cases you will want to manually configure the location of Java. For example:

```
export JAVA_HOME=/Library/Java/Home
export PATH="$PATH:$JAVA_HOME/bin"
```

if you're using bash or another variant of the Bourne Shell.

2.2 Downloading and Installing

The first step to getting up and running with Grails is to install the distribution. To do so follow these steps:

- Download a binary distribution of Grails and extract the resulting zip file to a location of your choice
- Set the GRAILS_HOME environment variable to the location where you extracted the zip
  - On Unix/Linux based systems this is typically a matter of adding something like the following `export GRAILS_HOME=/path/to/grails` to your profile
  - On Windows this is typically a matter of setting an environment variable under `My Computer/Advanced/Environment Variables`
- Then add the `bin` directory to your `PATH` variable.
  - On Unix/Linux based systems this can be done by adding `export PATH="$PATH:$GRAILS_HOME/bin"` to your profile
  - On Windows this is done by modifying the `Path` environment variable under `My Computer/Advanced/Environment Variables`

If Grails is working correctly you should now be able to type `grails -version` in the terminal window and see output similar to this:

```
Grails version: 2.0.0
```

2.3 Creating an Application

To create a Grails application you first need to familiarize yourself with the usage of the `grails` command which is used in the following manner:

```
grails [command name]
```

Run create-app to create an application:

```
grails create-app helloworld
```

GRAIL

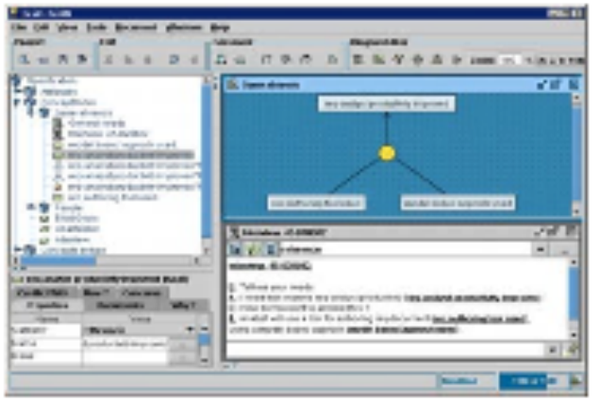
GRAIL[1] is a tool designed by RE practionners for RE practionners, to help them really **engineer requirements.** The tool relies on **KAOS**[2], the **goal-driven** requirements methodology. It helps industrial projects to succeed by **effectively** and **systematically eliciting** the requirements, defining system's agents and artefacts along with their expected behaviours. These elements are gathered, and linked together in a unique coherent model. GRAIL automates authoring process by **deriving** the requirements documents directly from the model.

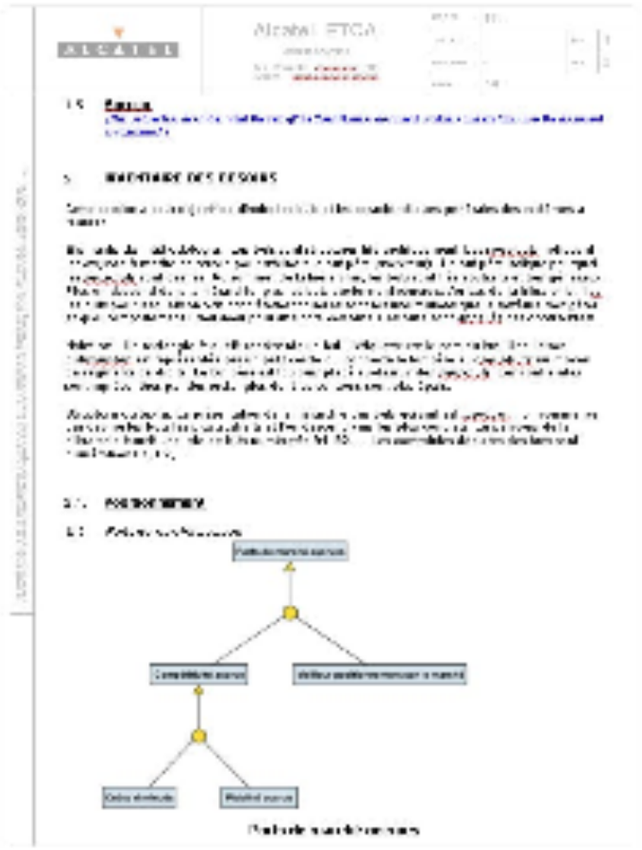
[1] More information on GRAIL at Cediti can be found at URL http://www.cediti.be/EN/Solutions_Services/requirements/

[2] More information on KAOS at UCL can be found at URL http://www.info.ucl.ac.be/research/projects/AVL//ReqEng.html

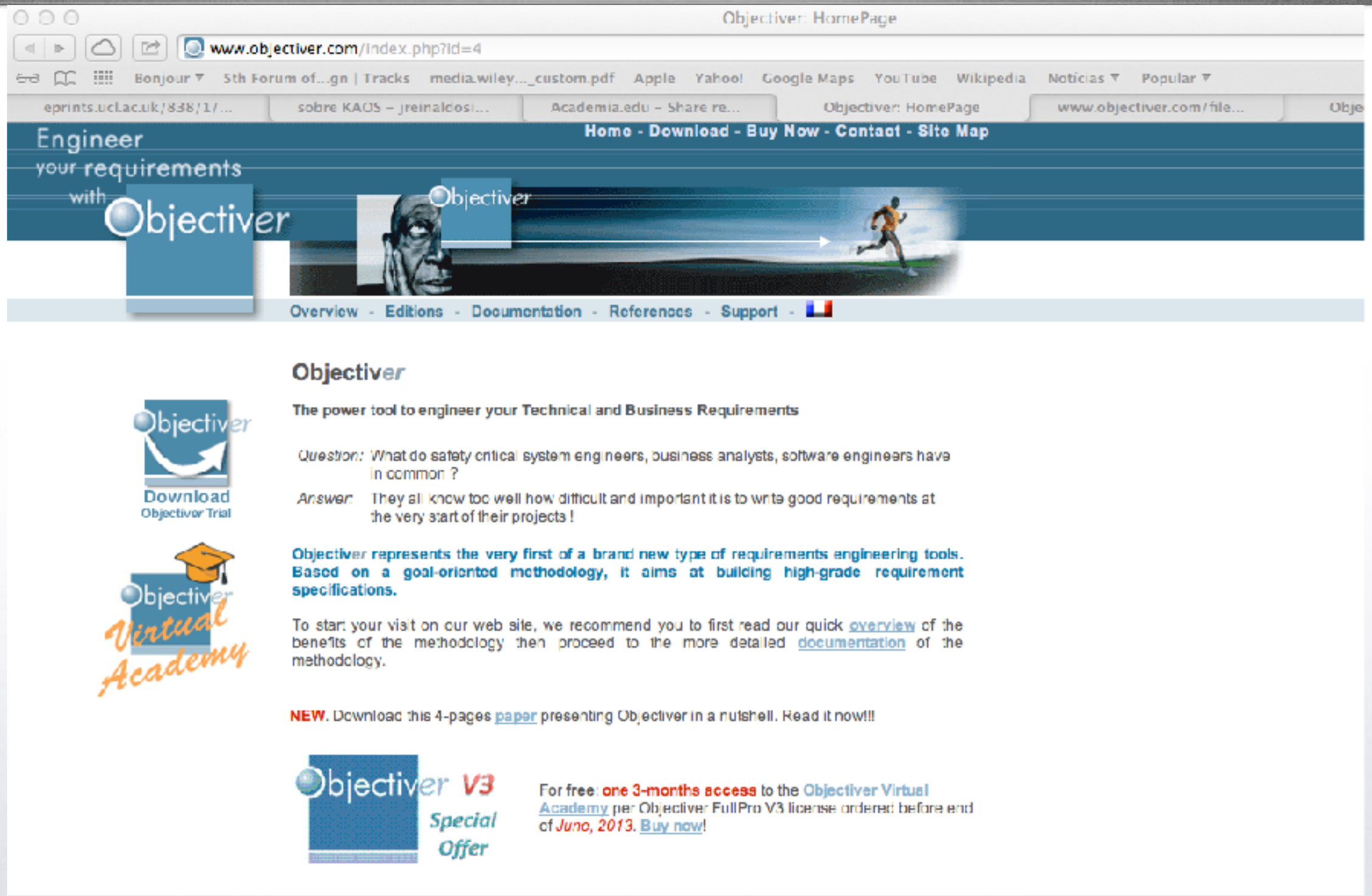

What ?
Who? When?
Why? On
How? what?
Kaos Model
Objective Refined
Analyst
Refine
Modelling


Corporate
Standard





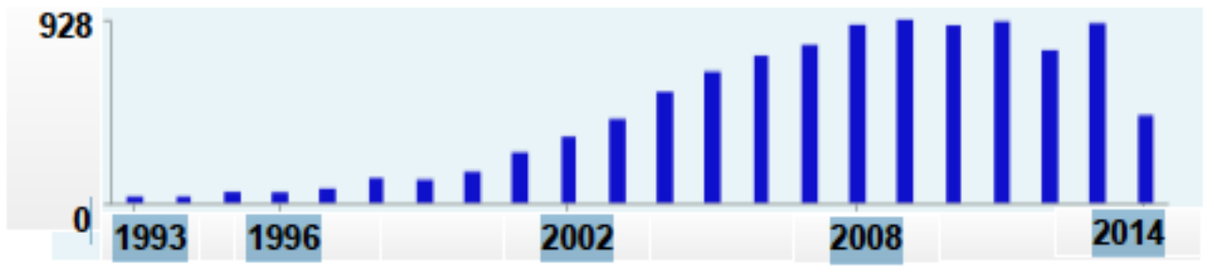
Grail

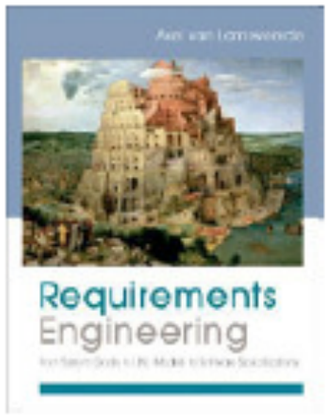

Objectiver: HomePage
www.objectiver.com/index.php?id=4
Bonjour
5th Forum of...gn | Tracks
media.wiley..._custom.pdf
Apple
Yahoo!
Google Maps
YouTube
Wikipedia
Noticias
Popular
eprints.ucl.ac.uk/838/1/...
sobre KAOS – jreinaldosi...
Academia.edu – Share re...
Objectiver: HomePage
www.objectiver.com/file...
Engineer
your requirements
with
Objectiver
Home - Download - Buy Now - Contact - Site Map
Overview - Editions - Documentation - References - Support -
Objectiver
The power tool to engineer your Technical and Business Requirements
Question: What do safety critical system engineers, business analysts, software engineers have in common ?
Answer: They all know too well how difficult and important it is to write good requirements at the very start of their projects !
Download
Objectiver Trial
Objectiver Virtual Academy
Objectiver represents the very first of a brand new type of requirements engineering tools. Based on a goal-oriented methodology, it aims at building high-grade requirement specifications.
To start your visit on our web site, we recommend you to first read our quick overview of the benefits of the methodology then proceed to the more detailed documentation of the methodology.
NEW. Download this 4-pages paper presenting Objectiver in a nutshell. Read it now!!!
Objectiver V3
Special Offer
For free: one 3-months access to the Objectiver Virtual Academy per Objectiver FullPro V3 license ordered before end of June, 2013. Buy now!

New Paradigm for requirements analysis

Axel Van Lamsweerde

	Todos	Desde 2009
Citações	10431	4872
Índice h	36	29
Índice i10	51	39





Requirements Engineering: From System Goals to UML Models to Software Specifications

Respect-IT

REquirements & **SPEC**ification **T**echniques for **I**nformation **T**echnology

Spin out of UCL (University of Louvain-Belgium).

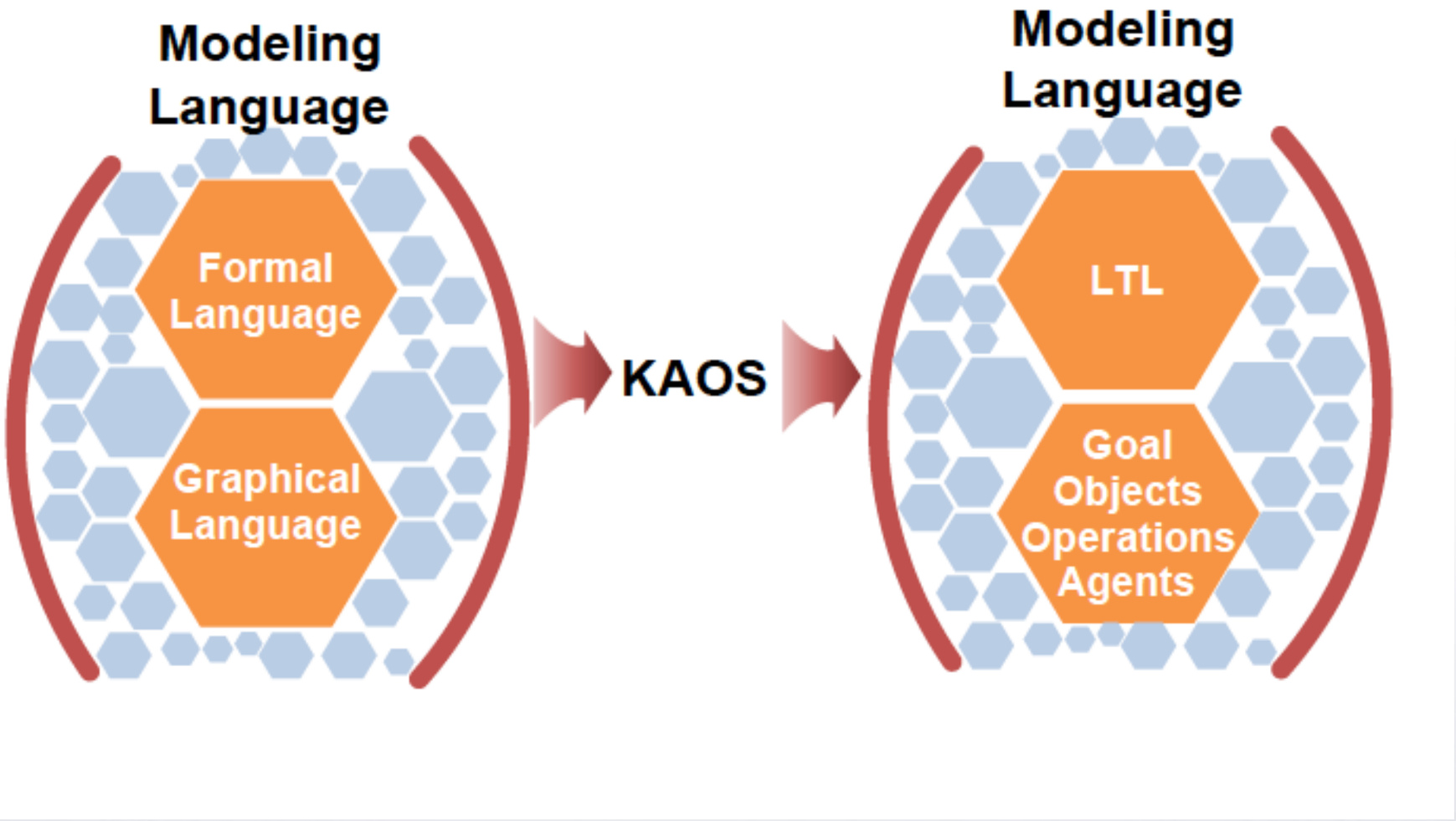
Scientific Advisor: Prof. Axel Van Lamswerde.

Main activities:

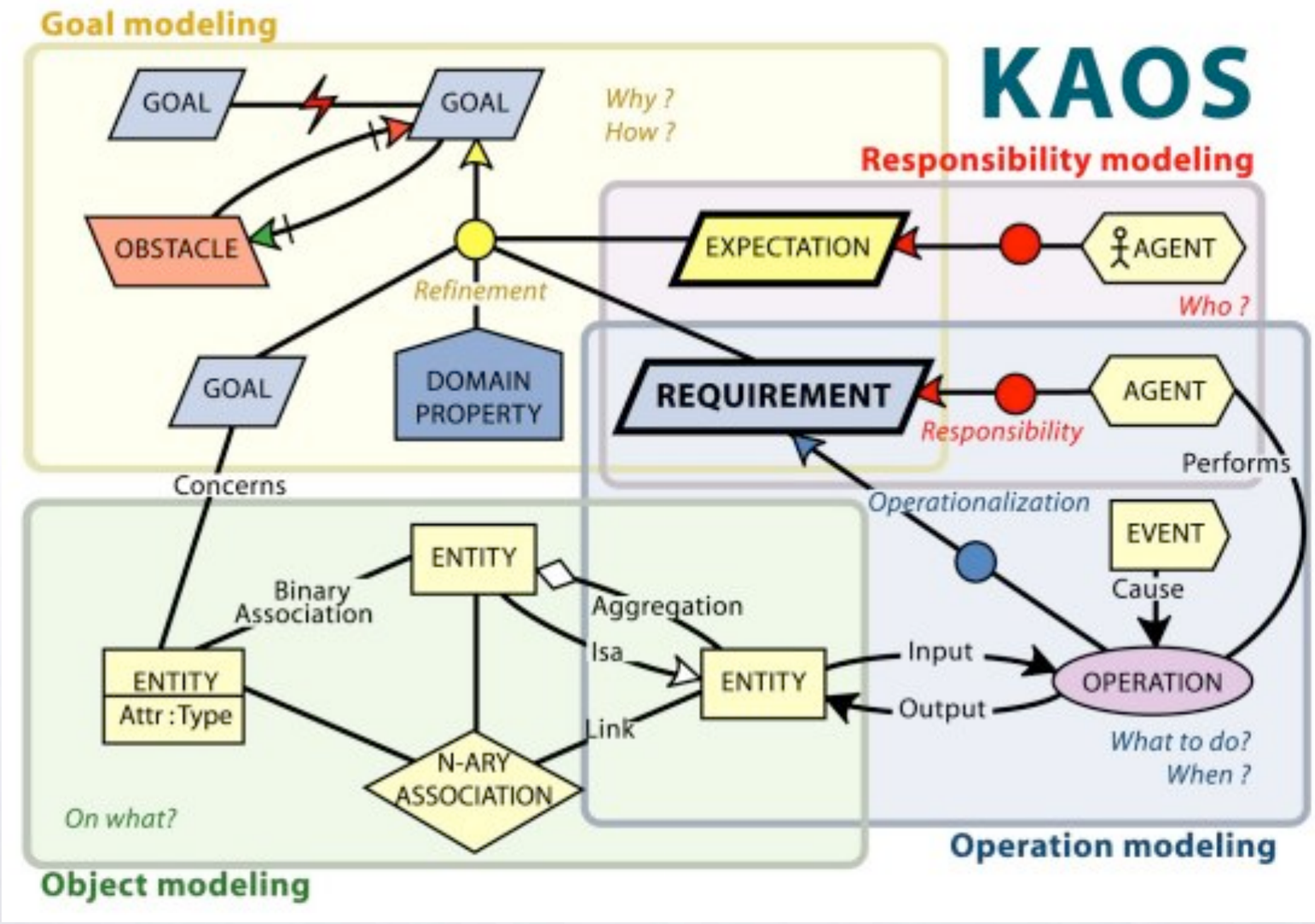
Consultancy service in GORE.

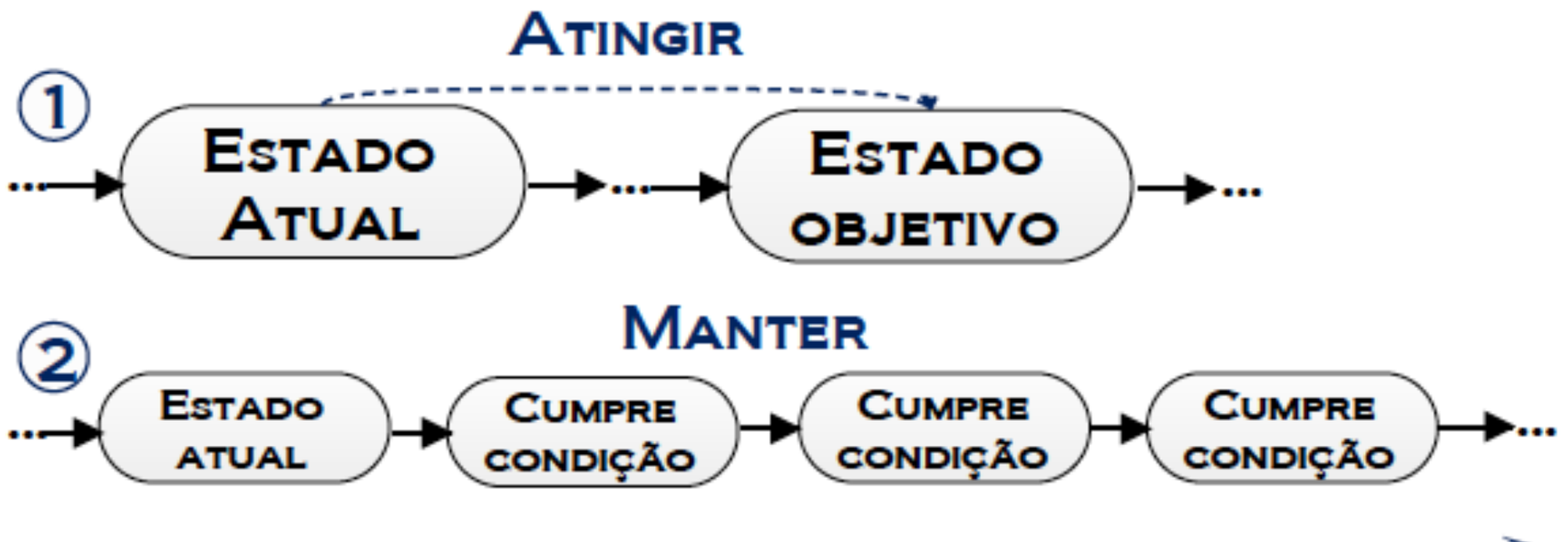
Tool Editor

Requirements modeling with KAOS



KAOS metamodel





Atingir [Estado Objetivo]
[if EstadoAtual then]
1 "eventualmente" EstadoObjetivo

Manter [CumpreCondição]
[if EstadoAtual then]
2 "sempre" CumpreCondição

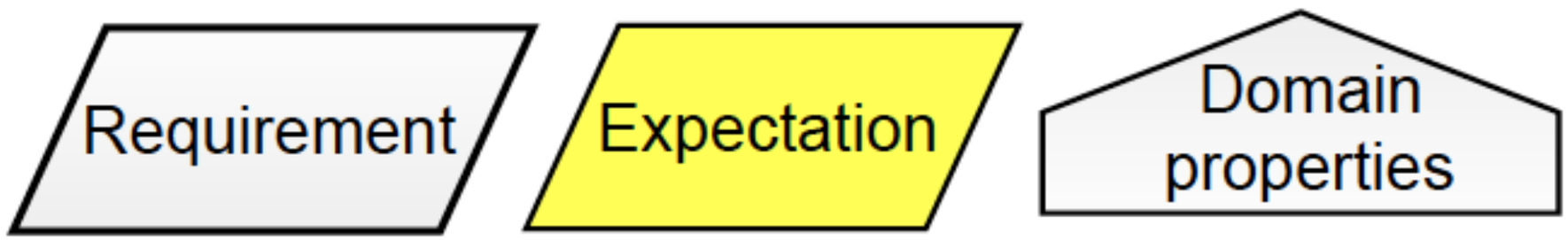
$$R, E, D \vdash G$$

"In view of properties D of the domain, the requirements R will achieve goals G under expectations E"

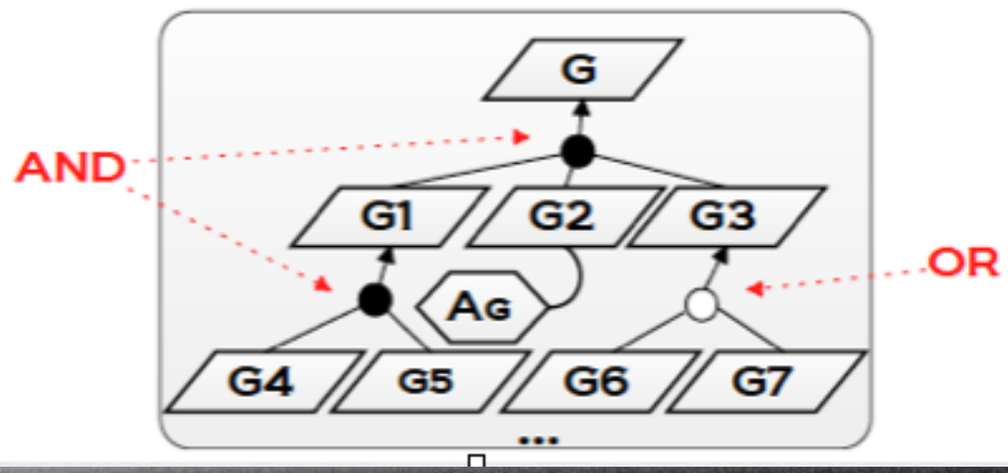
Requirement: Goal assigned to single agente in software-to-be.

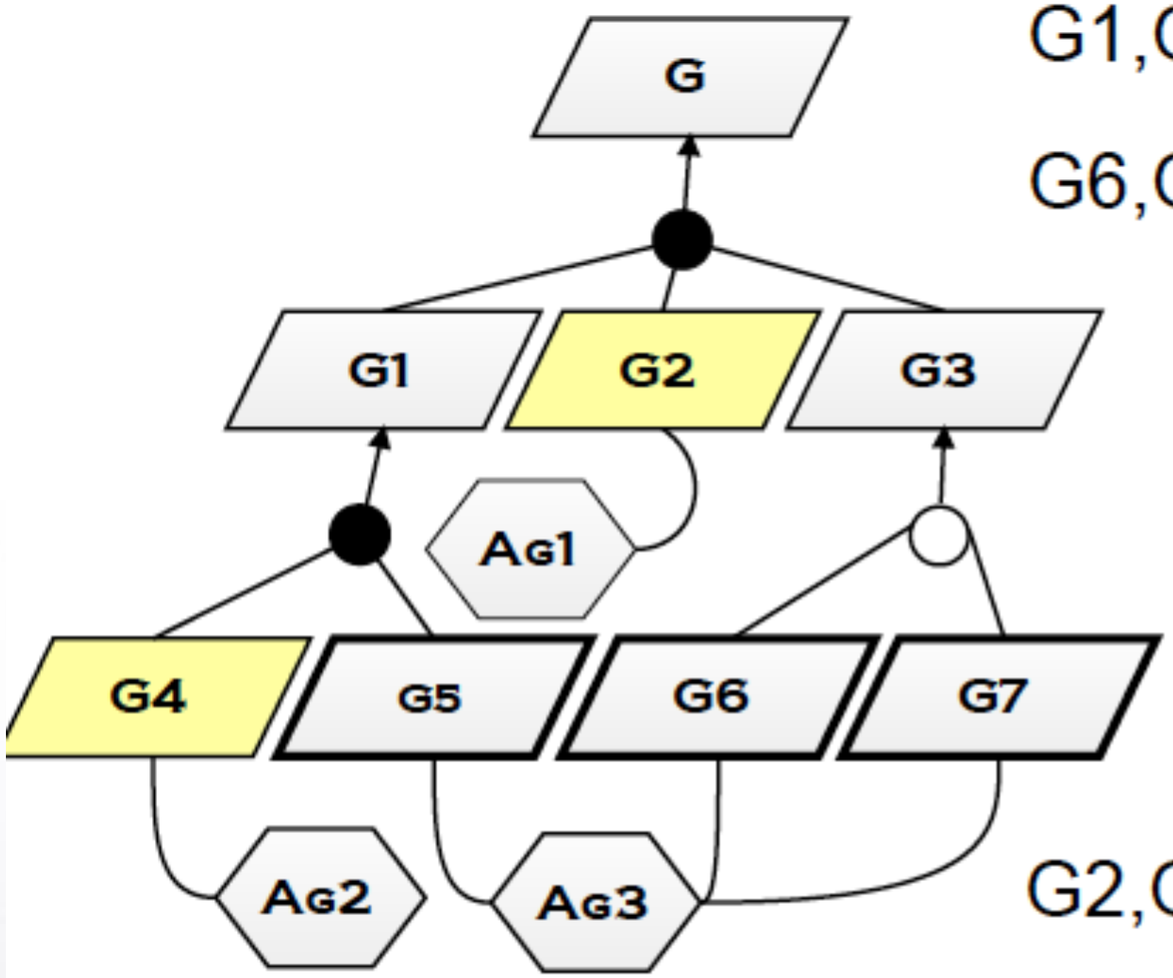
Expectation: Goal assigned to single agente in environment.

Domain properties: Descriptive statements about environment(physical laws, organizational policies)



- O refinamento AND do objetivo G nos sub-objetivos $G_1, ..., G_n$ expressa que G se cumpre, satisfazendo os sub-objetivos $G_1, ..., G_n$.
  - O conjunto é chamado de refinamento de G.
  - O sub-objetivo G_i é é uma contribuição positiva para G.
- O refinamento OR do objetivo G nos sub-objetivos $R_1, ..., R_m$ expressa que G se cumpre, satisfazendo pelo menos um sub-objetivo R_i do refinamento.
  - O(s) sub-objetivo(s) são chamados de alternativa(s) do refinamento.





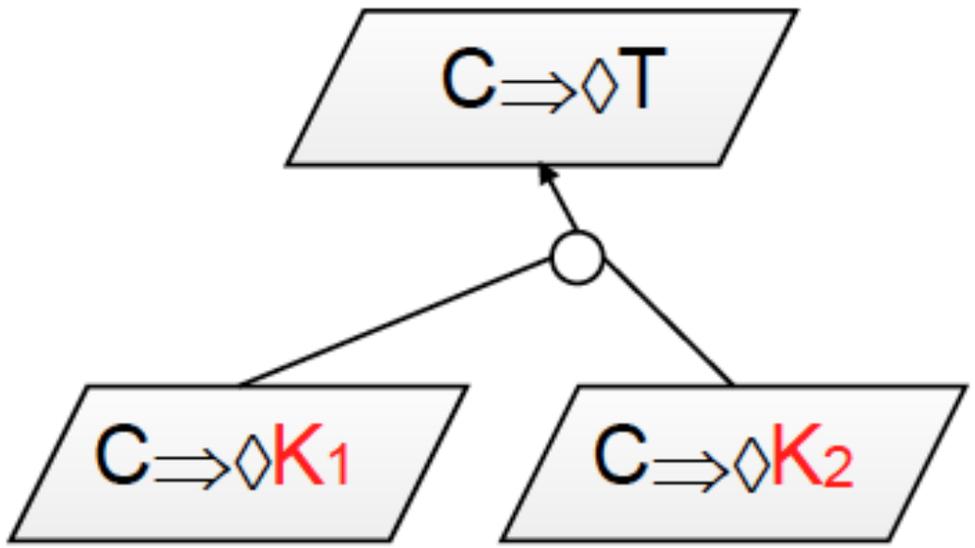
G1,G2,G3 **refinement** of G.

G6,G7 **alternatives** of G3.

G2,G4 **expectations** of agents Ag1 and Ag2.

G5,G6,G7 **requirements** of agents Ag3.

Case-Driven



$K_1 \vee K_2 \Rightarrow \Diamond C$: $K_1 \Rightarrow \Diamond C$, $\vee K_2 \Rightarrow \Diamond C$. Then:

$K_1 \Rightarrow \Diamond T$, $K_2 \Rightarrow \Diamond T$.

Rule generalization

$K_1 \vee K_2 \vee \ldots \vee K_n \Rightarrow \Diamond C$:

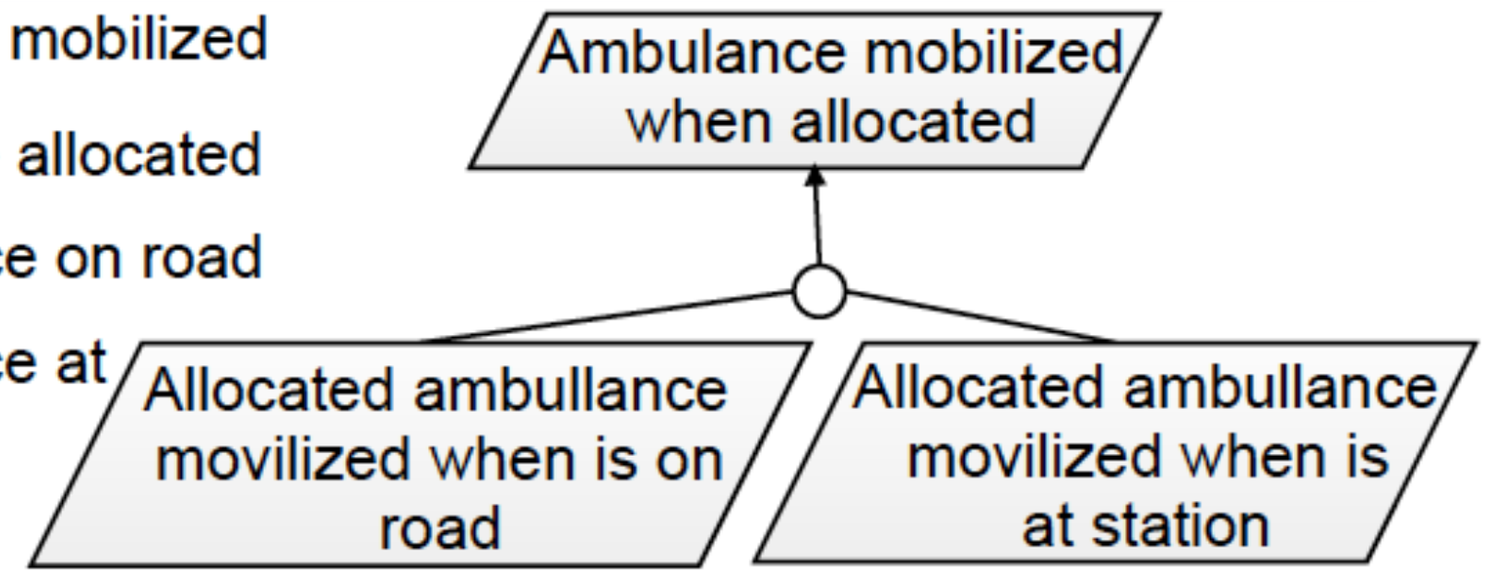
$K_1 \vee K_2 \vee \ldots \vee K_n \Rightarrow \Diamond T$.

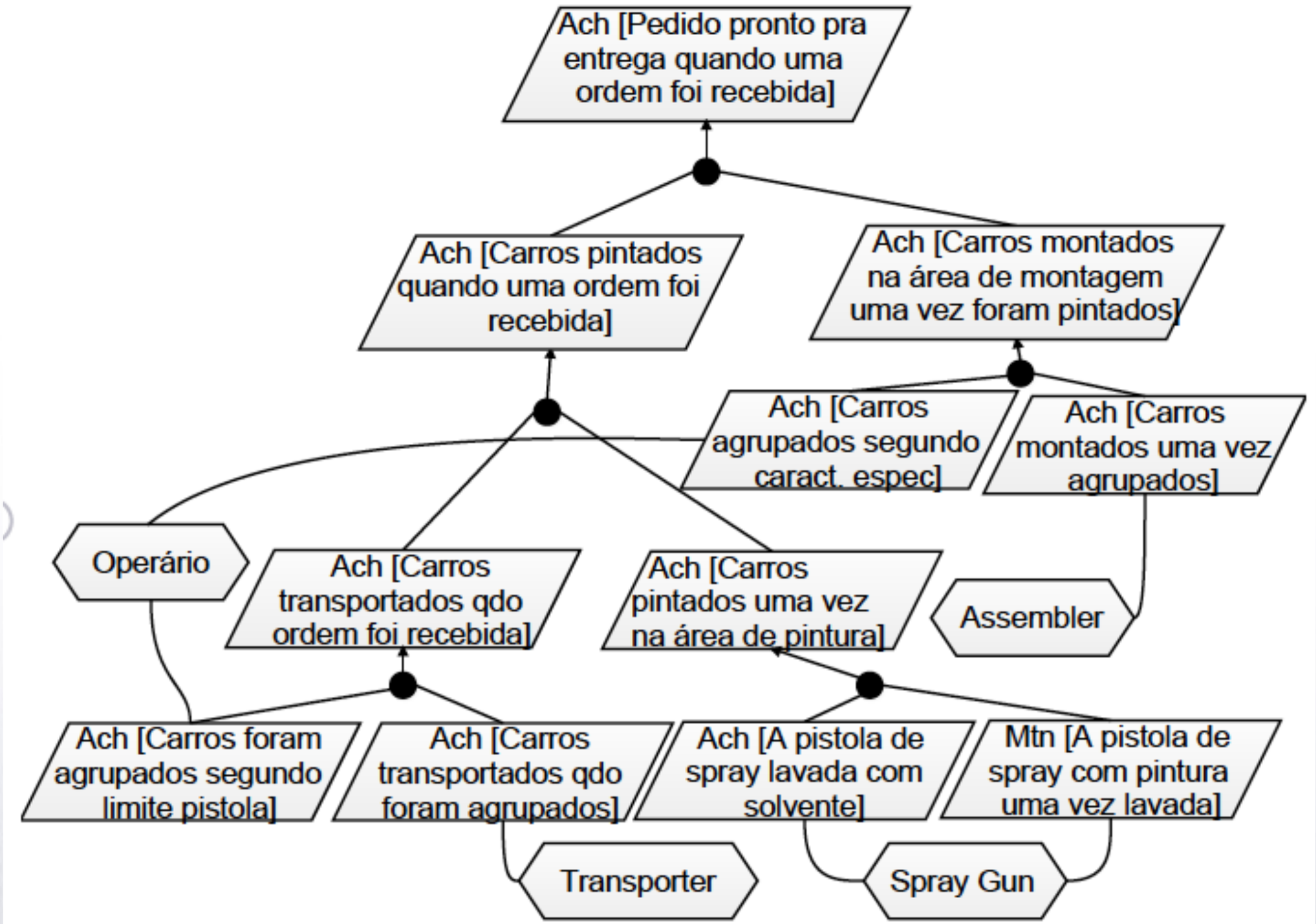
T:Ambullance mobilized

C:Ambullance allocated

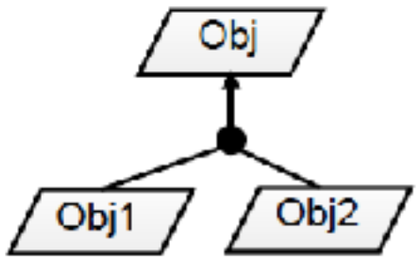
K_1: Ambullance on road

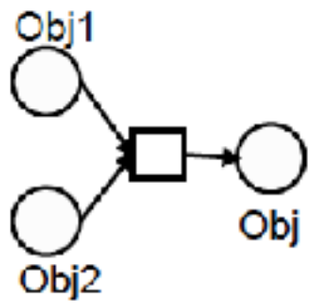
K_2: Ambullance at station.



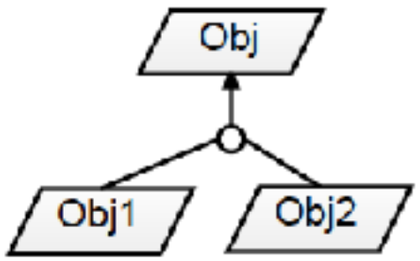

Ach [Pedido pronto pra entrega quando uma ordem foi recebida]
Ach [Carros pintados quando uma ordem foi recebida]
Ach [Carros montados na área de montagem uma vez foram pintados]
Ach [Carros agrupados segundo caract. espec]
Ach [Carros montados uma vez agrupados]
Operário
Ach [Carros transportados qdo ordem foi recebida]
Ach [Carros pintados uma vez na área de pintura]
Assembler
Ach [Carros foram agrupados segundo limite pistola]
Ach [Carros transportados qdo foram agrupados]
Ach [A pistola de spray lavada com solvente]
Mtn [A pistola de spray com pintura uma vez lavada]
Transporter
Spray Gun

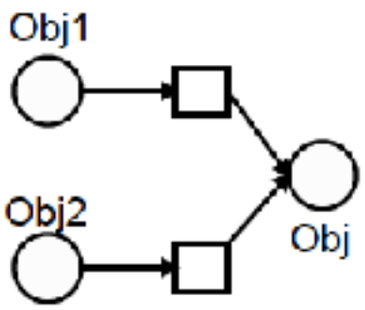
Refinamento-and



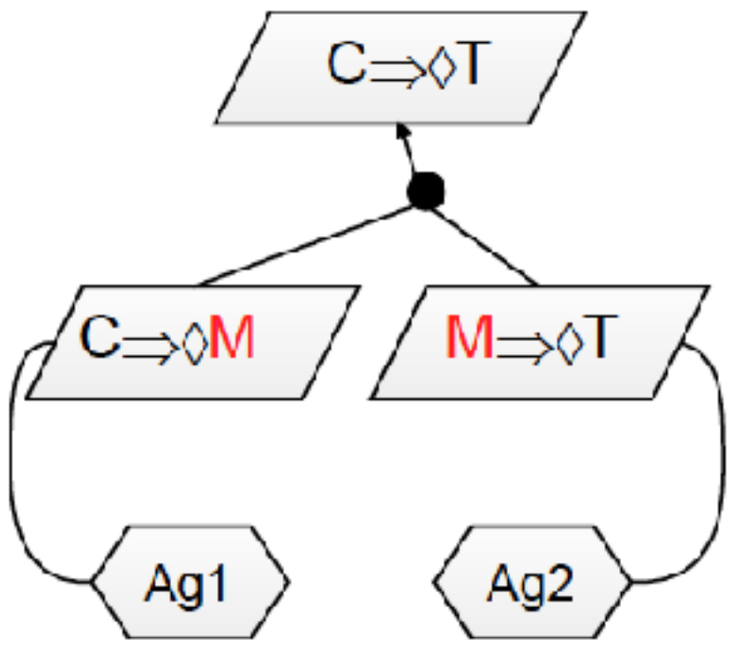


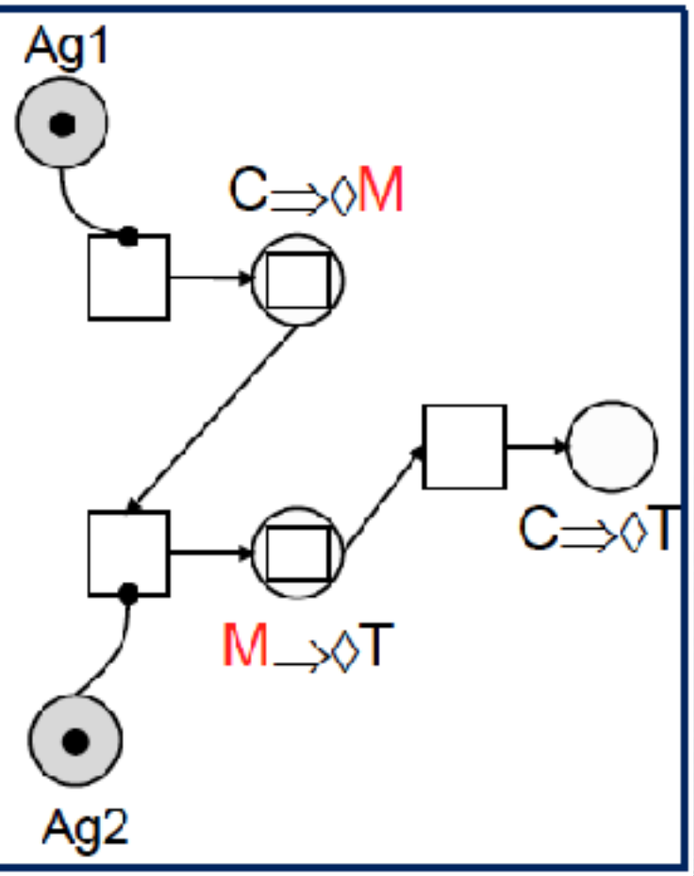
Refinamento-or





MILESTONE-DRIVE



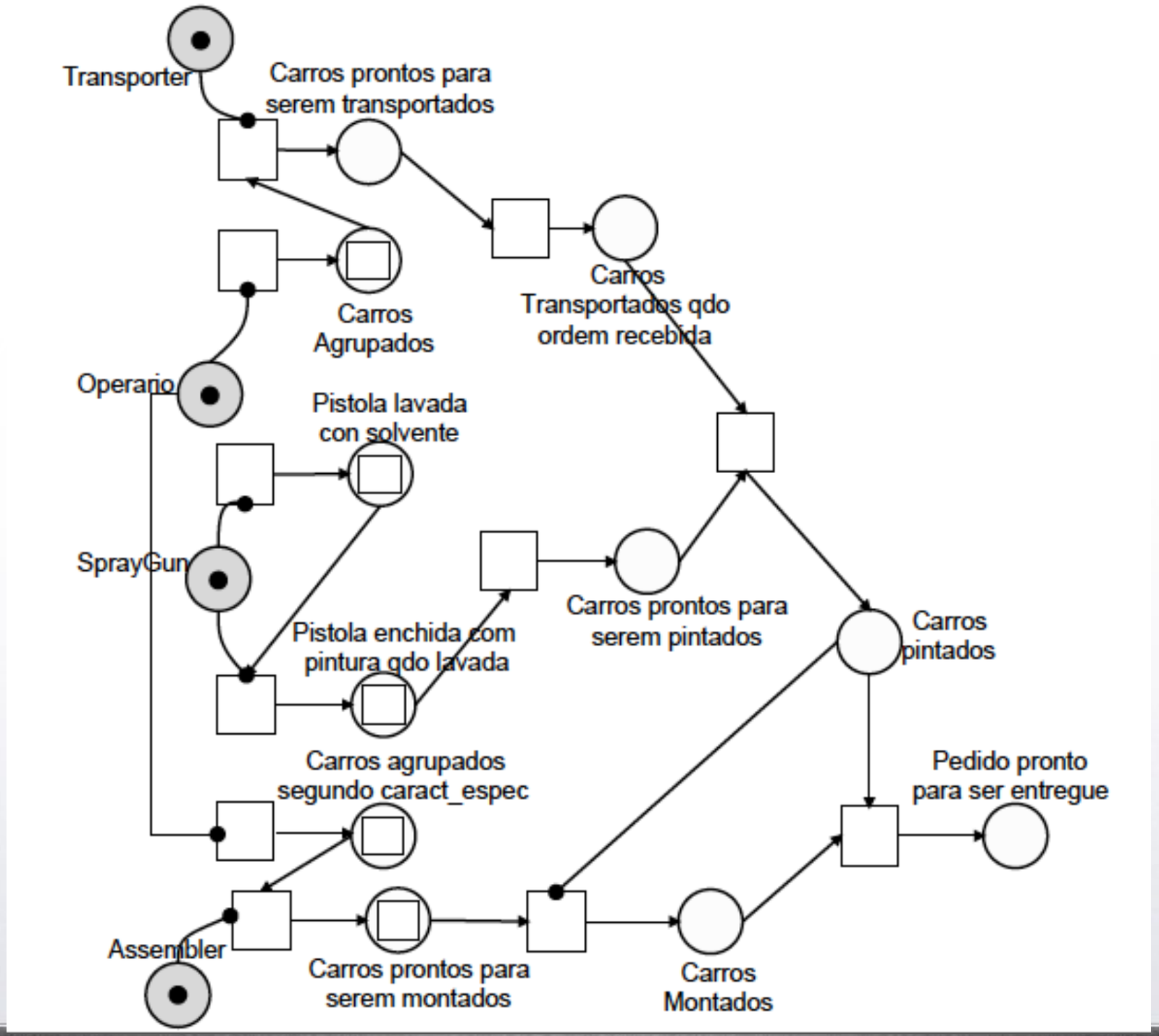


The goals take the general formal form:

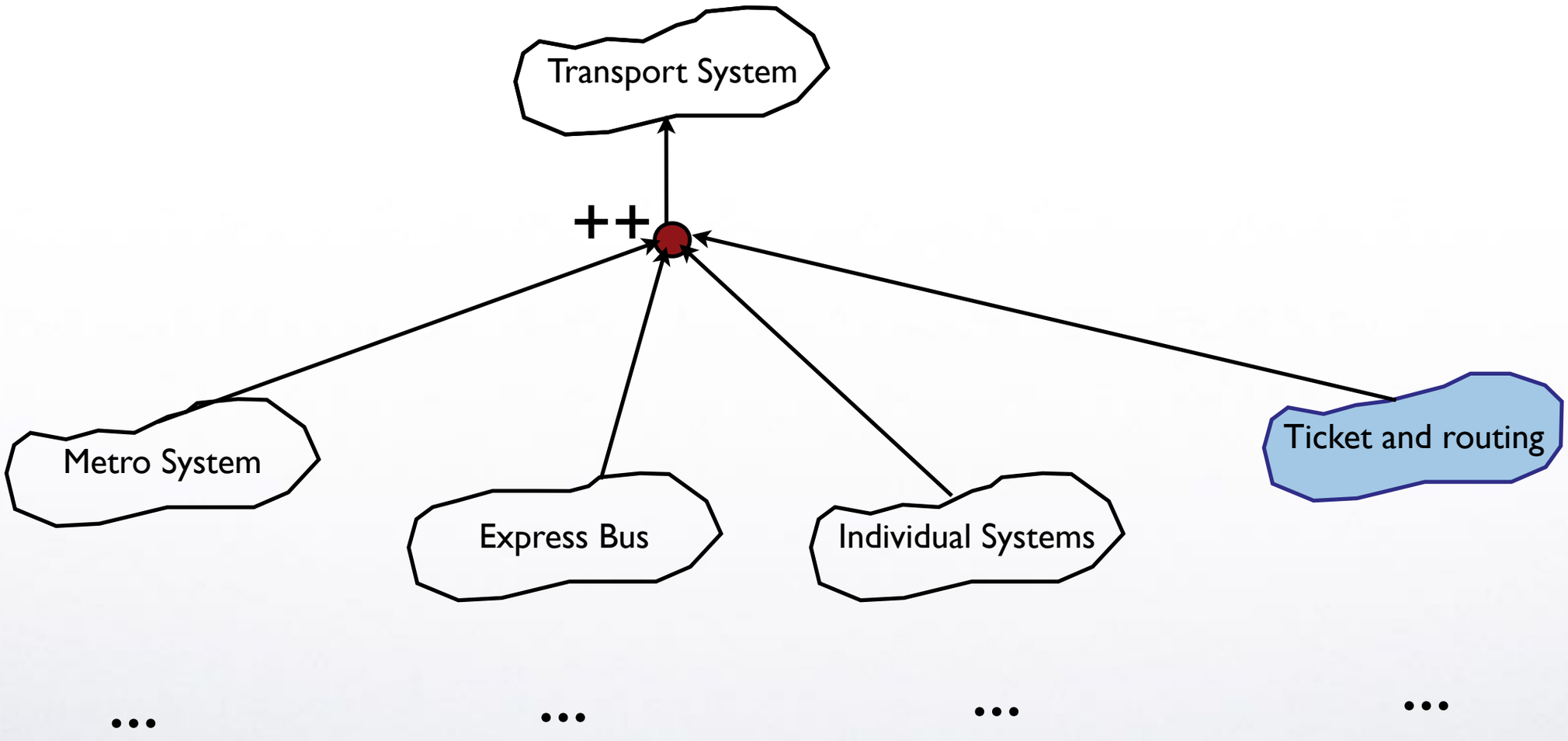
$$C \Rightarrow \Theta T$$

where:

- C is the current condition
- T is the target condition
- Θ represents a LTL operator such as:
  - $\bigcirc$: In the next state.
  - $\Diamond$: Sometimes in the future. [$\Diamond_{\leq d}$: Sometimes in the future before deadline d].
  - $\Box$: Always in the future. [$\Box_{\leq d}$: Always in the future up to deadline d]


Transporter
Carros prontos para serem transportados
Carros Agrupados
Carros Transportados qdo ordem recebida
Operario
Pistola lavada con solvente
SprayGun
Pistola enchida com pintura qdo lavada
Carros prontos para serem pintados
Carros pintados
Carros agrupados segundo caract_espec
Pedido pronto para ser entregue
Assembler
Carros prontos para serem montados
Carros Montados

Kaos and the system of system approach



Exemplo BART (Bay Area Rapid Transit)

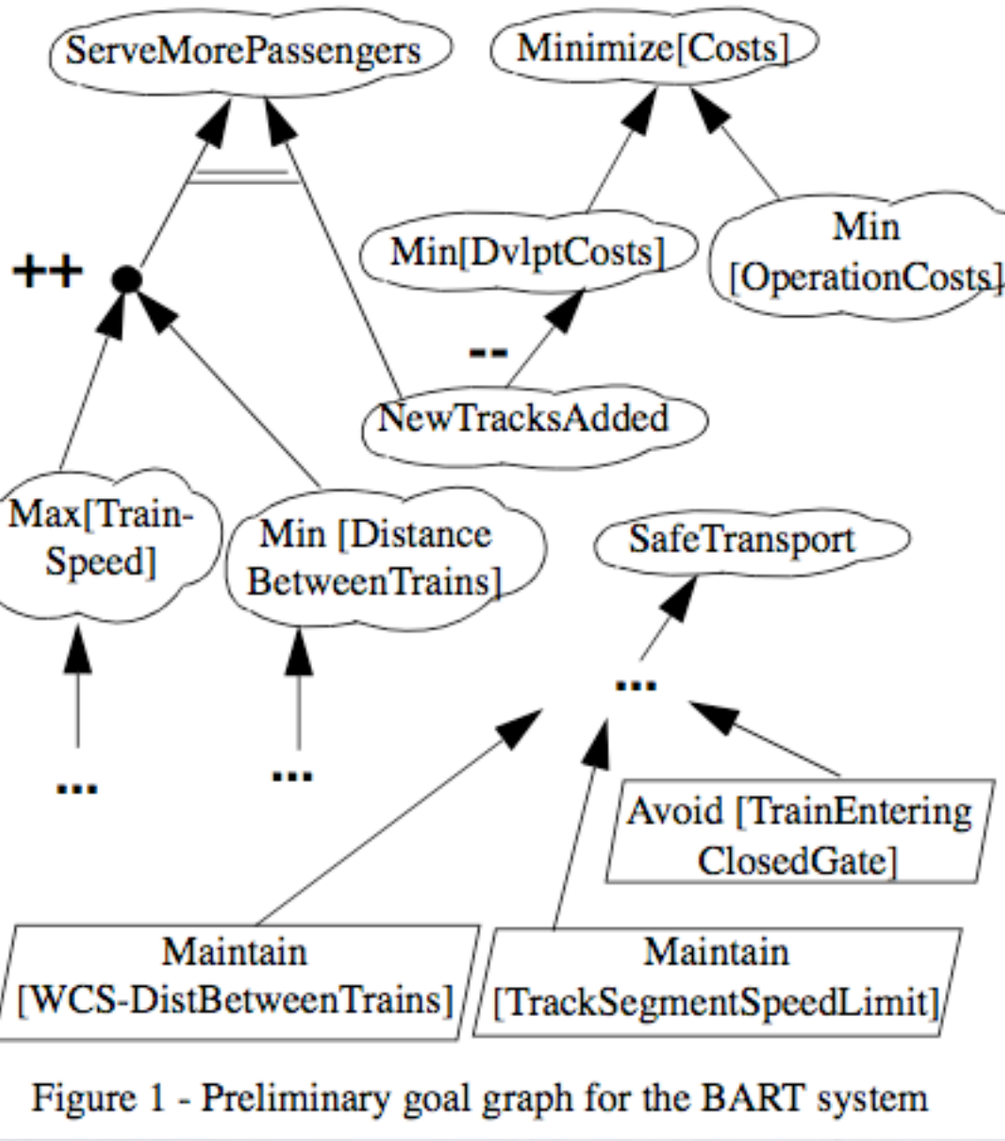
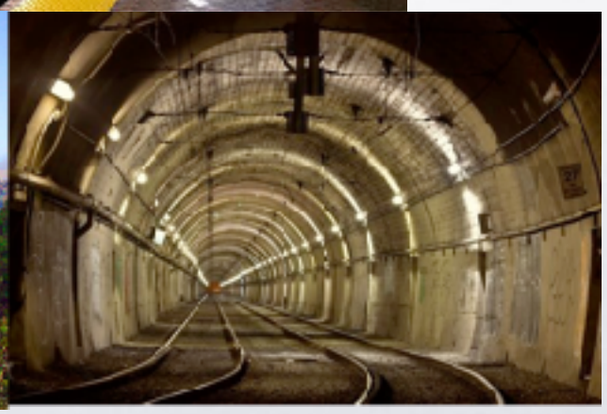
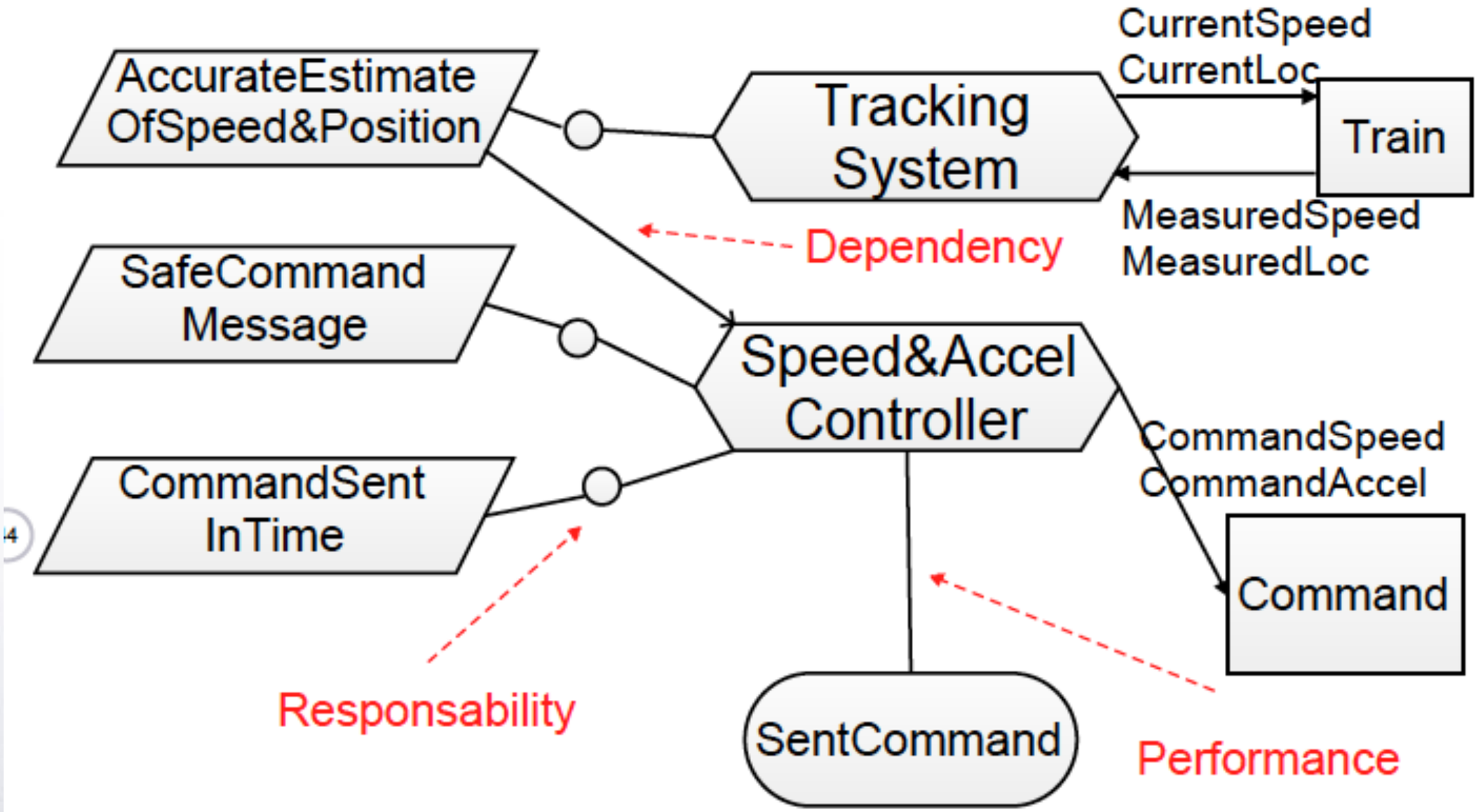


Figure 1 - Preliminary goal graph for the BART system




AccurateEstimate
OfSpeed&Position
SafeCommand
Message
CommandSent
InTime
Tracking
System
Dependency
Speed&Accel
Controller
CurrentSpeed
CurrentLoc
Train
MeasuredSpeed
MeasuredLoc
CommandSpeed
CommandAccel
Command
SentCommand
Responsability
Performance

Detectando conflitos

Conflitos podem ser detctados e representados graficamente localizando os pontos de negociação direto na documentação de forma mais explícita e expressiva.

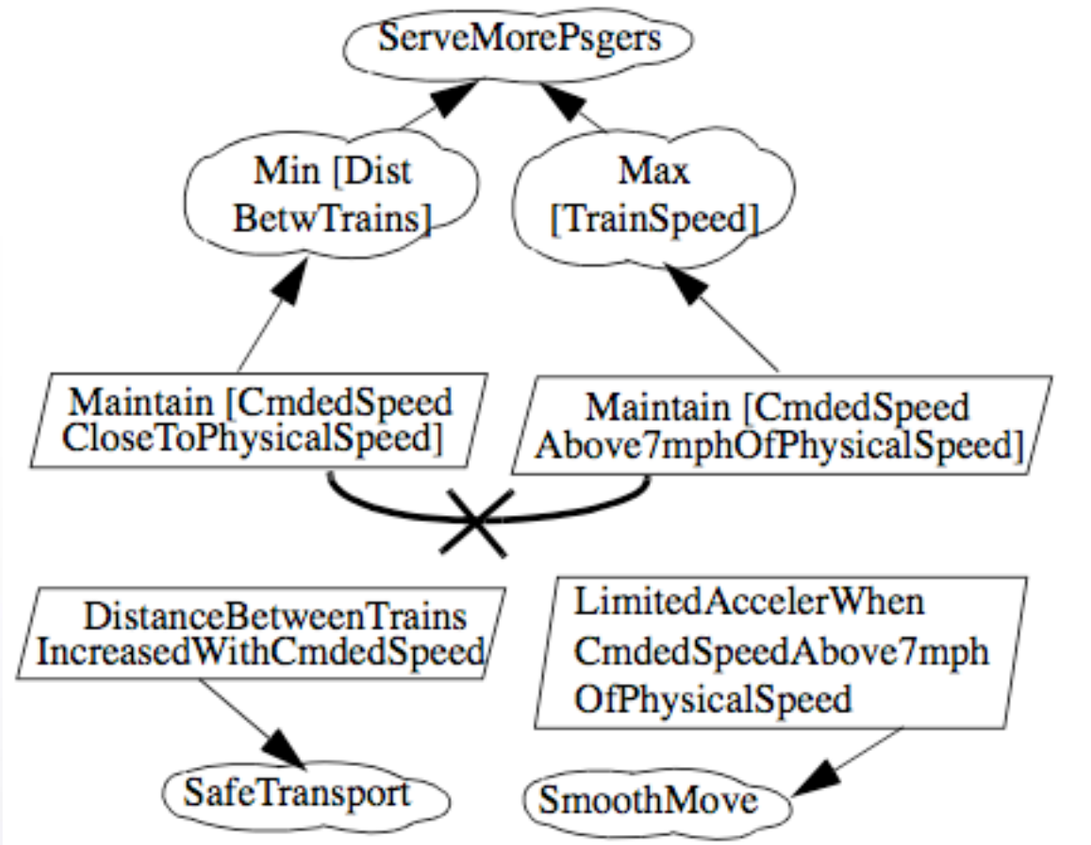


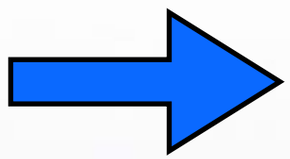
Figure 4 - Conflict in speed/acceleration control

Resolvendo conflitos

Goal Maintain [CmdedSpeedAbove7mphOfPhysicalSpeed]

FormalDef ∀ tr: Train

$$tr.Acc_{CM} \geq 0 \Rightarrow tr.Speed_{CM} > tr.Speed + 7$$



Goal Maintain [CmdedSpeedAbove7mphOfPhysicalSpeed]

FormalDef ∀ tr: Train

$$tr.Acc_{CM} \geq 0 \rightarrow tr.Speed_{CM} > tr.Speed + 7$$
$$\vee\ f\,(dist\text{-}to\text{-}obstacle) \leq 7$$

A documentação

GOAL *TipoComportamento NomeObjetivo*

DEF *Descrição textual informal do objetivo*

[**FORMALSPEC** : [*Sentenças LTL*]

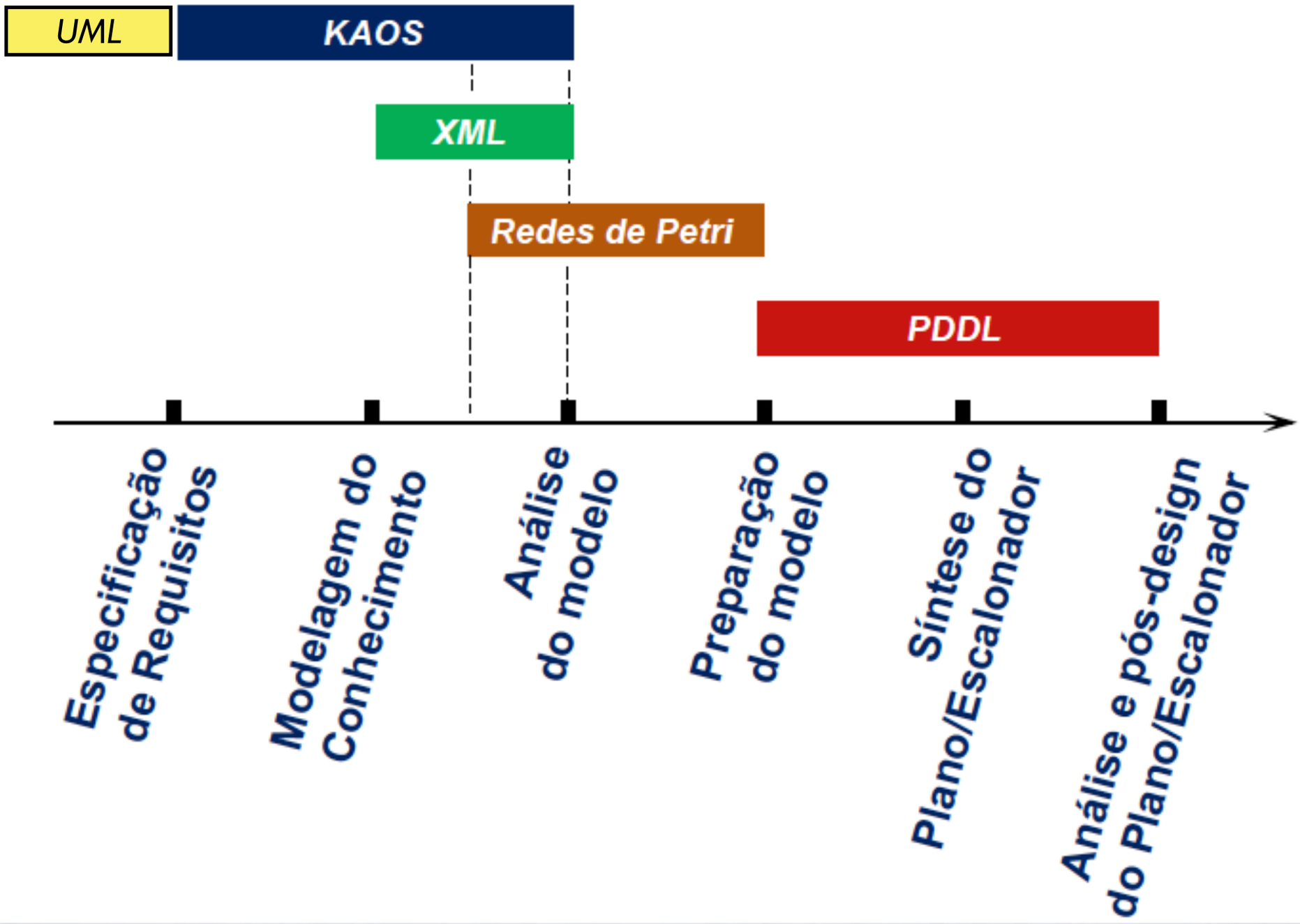
EXEMPLO:

Goal Achieve CarroPintadoQdoTransportado

Def **A carro ficará pintado em algum momento uma vez na área de pintura.**

FormalSpec $\forall c : Carro, ap : AreaPintura$

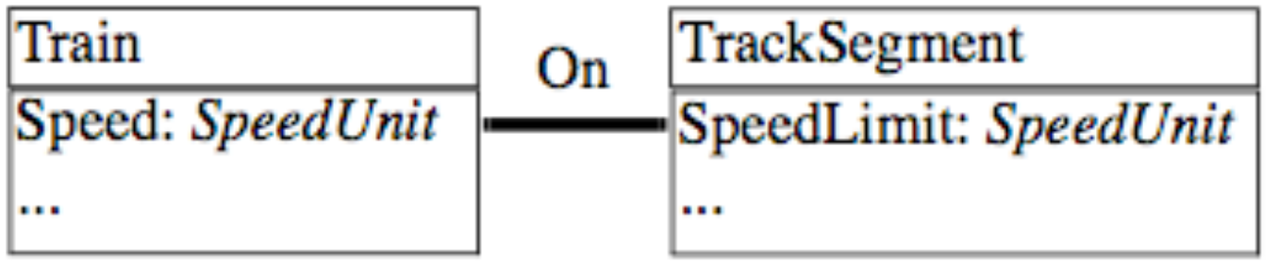
$$Em(c, ap) \Rightarrow \Diamond c.pintado = 'true'$$


UML
KAOS
XML
Redes de Petri
PDDL
Especificação de Requisitos
Modelagem do Conhecimento
Análise do modelo
Preparação do modelo
Síntese do Plano/Escalonador
Análise e pós-design do Plano/Escalonador

Refinando e formalizando goals (objetivos)

Goal Maintain[WCS-DistBetweenTrains]
InformalDef *A train should never get so close to a train in front so that if the train in front stops suddenly (e.g., derailment) the next train would hit it.*
FormalDef ∀ tr1, tr2: Train :
Following(tr1, tr2) ⇒ tr1.Loc - tr2.Loc > tr1.WCS-Dist

Regra de segurança



Goal Maintain[TrackSegmentSpeedLimit]
InformalDef *A train should stay below the maximum speed the track segment can handle.*
FormalDef ∀ tr: Train, s: TrackSegment :
On(tr, s) → tr.Speed ≤ s.SpeedLimit

Segunda fase da modelagem de requisitos

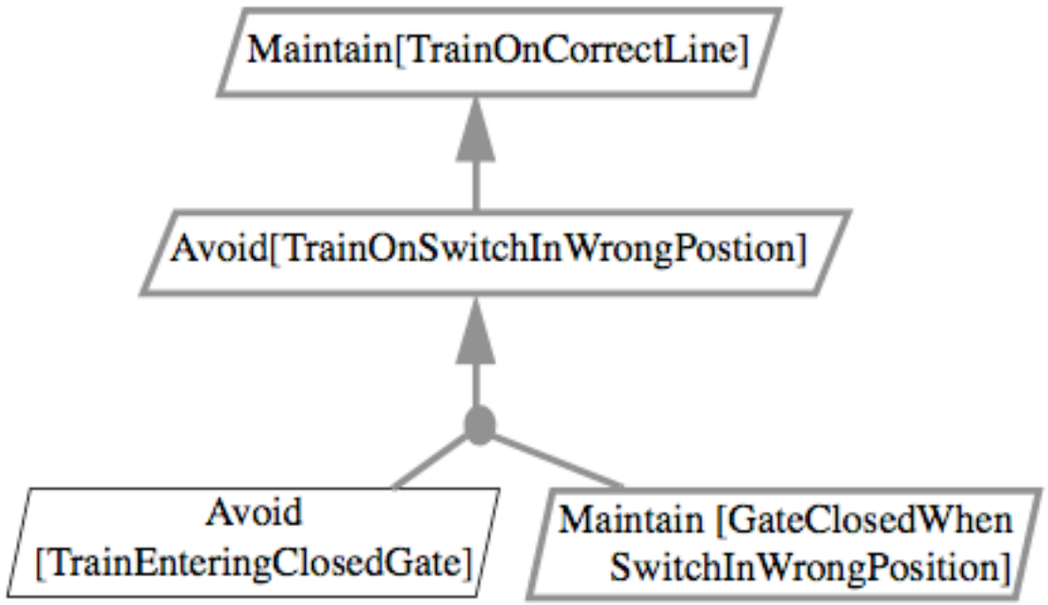


Figure 2 - Enriching the goal graph by WHY elicitation

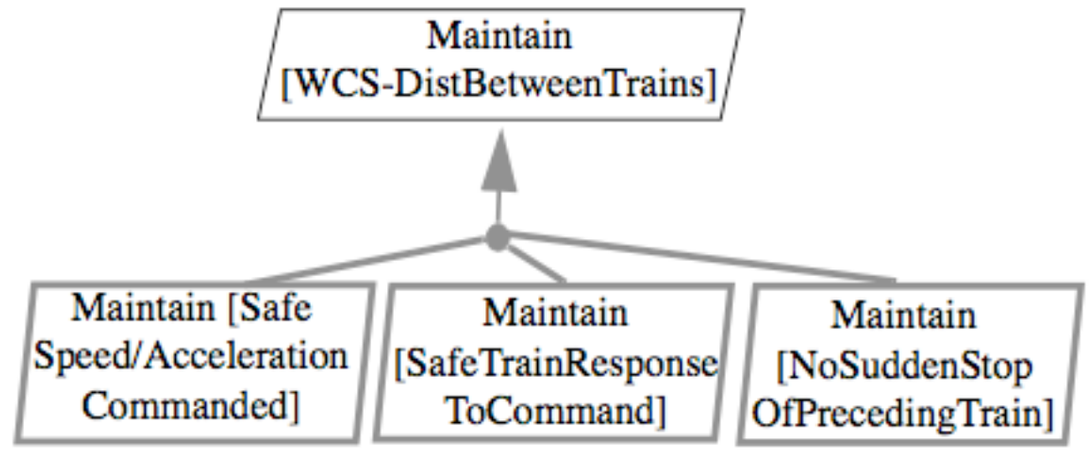


Figure 3 - Goal refinement

A base da modelagem está na relação WHAT-WHY, onde se introduz o conceito de *rationales*, isto é porque as decisões são tomadas.

Atribuindo respnsabilidades

O agente responsável pelo goal Maintain[SafeCmdMsg] é o controle de aceleração e velocidade Acel/SpeedControl, que por sua vez tem a seguinte interface de ação sobre as variáveis cm.Accel e cm.Speed:

Goal Maintain[SafeCmdMsg]

FormalDef $\forall$ cm: *CommandMessage*, ti1, ti2: *TrainInfo*

$$cm.Sent \wedge cm.TrainID = ti1.TrainID \wedge FollowingInfo\ (ti1, ti2)$$
$$\Rightarrow cm.Accel \leq F\ (ti1, ti2) \wedge cm.Speed > G\ (ti1)$$

Operacionalizando Goals

Operation SendCommandMessage
Input Train {**arg** tr}, TrainInfo; **Output** ComandMsg {**res** cm}
DomPre ... ; **DomPost** ...
ReqPost for SafeCmdMsg:
Tracking (ti1, tr) $\wedge$ Following (ti1, ti2)
$\rightarrow$ cm.Acc $\leq$ F (ti1, ti2) $\wedge$ cm.Speed $>$ G (ti1)
ReqTrig for CmdMsgSentInTime:
$\blacksquare_{\leq 0.5\ sec}$ $\neg$ $\exists$ cm2: CommandMessage:
cm2.Sent $\wedge$ cm2.TrainID = tr.ID

A Nova Fase do Curso

Chegamos no segundo módulo do curso onde faremos uma mudança estratégica passando de uma fase de discussão mais conceitual para uma fase mais prática, visando a aquisição de habilidades juntamente com os novos conceitos.

Mini-projeto

Fase 1: Modelagem de um sistema automatizado para controle de equipamento dos laboratórios de pesquisa em KAOS (individual); deadline: 1 de novembro, à meia-noite.

Nesse sistema todo o material do laboratório (software, hardware, livros, documentos, etc.) seriam identificados por número de série (hardware), licença (software), número de tomo (livros e documentos). Eventualmente, alguns deles podem ter RFID e o laboratório deve ter um leitor na porta de entrada.

Um sistema de banco de dados deve guardar toda a movimentação de entrada e saída de material do laboratório e o responsável pela guarda. Eventualmente uma saída sem registro poderá ser identificada pelo RFID (se o item possui RFID) e com a imagem guardada na hora da saída. Um operador deve então fazer o registro à revelia.

Tês movimentações à revelia suspende o usuário que fica então impedido de fazer novas movimentações. O tempo que cada item pode estar fora depende do item e da atividade. O sistema deve então enviar mensagens (ou e-mails) solicitando a devolução ou renovação da saída. O equipamento deve necessariamente VOLTAR (fisicamente) para gerar uma nova movimentação.

Leitura da semana

Goal-Oriented Requirements Engineering: A Guided Tour

Axel van Lamsweerde

Département d'Ingénierie Informatique
Université catholique de Louvain
B-1348 Louvain-la-Neuve (Belgium)
avl@info.ucl.ac.be

Abstract

Goals capture, at different levels of abstraction, the various objectives the system under consideration should achieve. Goal-oriented requirements engineering is concerned with the use of goals for eliciting, elaborating, structuring, specifying, analyzing, negotiating, documenting, and modifying requirements. This area has received increasing attention over the past few years.

The paper reviews various research efforts undertaken along this line of research. The arguments in favor of goal orientation are first briefly discussed. The paper then compares the main approaches to goal modeling, goal specification and goal-based reasoning in the many activities of the requirements engineering process. To make the discussion more concrete, a real case study is used to suggest what a goal-oriented requirements engineering method may look like. Experience with such approaches and tool support are briefly discussed as well.

1. Introduction

Goals have long been recognized to be essential components involved in the requirements engineering (RE) process. As Ross and Schoman stated in their seminal paper, *"requirements definition must say why a system is needed, based on current or foreseen conditions, which may be internal operations or an external market. It must say what system features will serve and satisfy this context. And it must say how the system is to be constructed"* [Ros77]. Many informal system development methodologies from the good old times included some form of goal-based analysis and from the literature on object-oriented analysis (one notable exception is [Rub92]). UML advocates sometimes confess the need for higher-level abstractions: *"In my work, I focus on user goals first, and then I come up with use cases to satisfy them: by the end of the elaboration period, I expect to have at least one set of system interaction use cases for each user goal I have identified"* [Fou97, p.45]. The prominent tendency in software modeling research has been to abstract programming constructs up to requirements level rather than propagate requirements abstractions down to programming level [Myl99].

Requirements engineering research has increasingly recognized the leading role played by goals in the RE process [Yue87, Rob89, Der91, Dar91, Myl92, Jar93, Zav97b]. Such recognition has led to a whole stream of research on goal modeling, goal specification, and goal-based reasoning for multiple purposes, such as requirements elaboration, verification or conflict management, and under multiple forms, from informal to qualitative to formal.

The objective of this paper is to provide a brief but hopefully comprehensive review of the major efforts undertaken along this line of research. Section 2 first provides some background material on what goals are, what they are useful for, where they are coming from, and when they should be made explicit in the RE process. Section 3 discusses the major efforts in modeling goals in terms of features and links to other artefacts found in requirements models. Section 4 reviews the major techniques used for specifying goals. Section 5 on goal-based reasoning reviews how goals are used in basic activities of the RE process such as

Obrigado

Reinaldo