

Evaluating and Creatively Building KAOS Goal Models

João Araújo

joao.araujo@fct.unl.pt, <http://ctp.di.fct.unl.pt/~ja/>

2

1st Part: Evaluating KAOS Models

In collaboration with Patricia Espada and Miguel Goulão (presented at CAiSE 2013)

Goal-Oriented Requirements Engineering (GORE)

3

Espada, Goulão, Araújo, "A Framework to Evaluate Complexity and Completeness of KAOS Goal Models", CAiSE 2013, Valencia, Spain

- A paradigm in Requirements Engineering to handle
 - ▣ Requirements **elicitation**
 - ▣ Requirements **specification**
 - ▣ Requirements **analysis**
 - ▣ Requirements **negotiation**
 - ▣ Requirements **evolution**

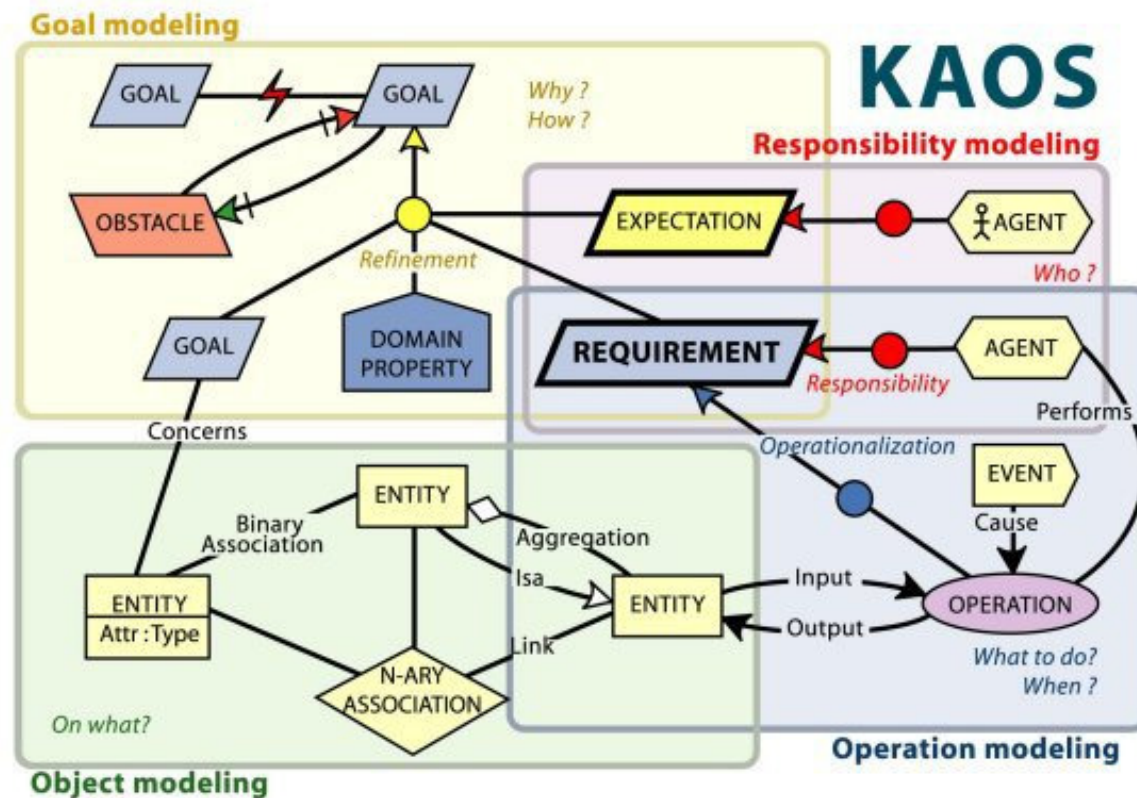
- Some well-known approaches
 - ▣ **KAOS**, i* framework, GBRAM, GRL, ...

Knowledge Acquisition in autOmated Specification (KAOS)

4

Espada, Goulão, Araújo, "A Framework to Evaluate Complexity and Completeness of KAOS Goal Models", CAiSE 2013, Valencia, Spain

- GORE methodology based in goal decomposition and refinement, to support requirements acquisition and elaboration



Let's work out...

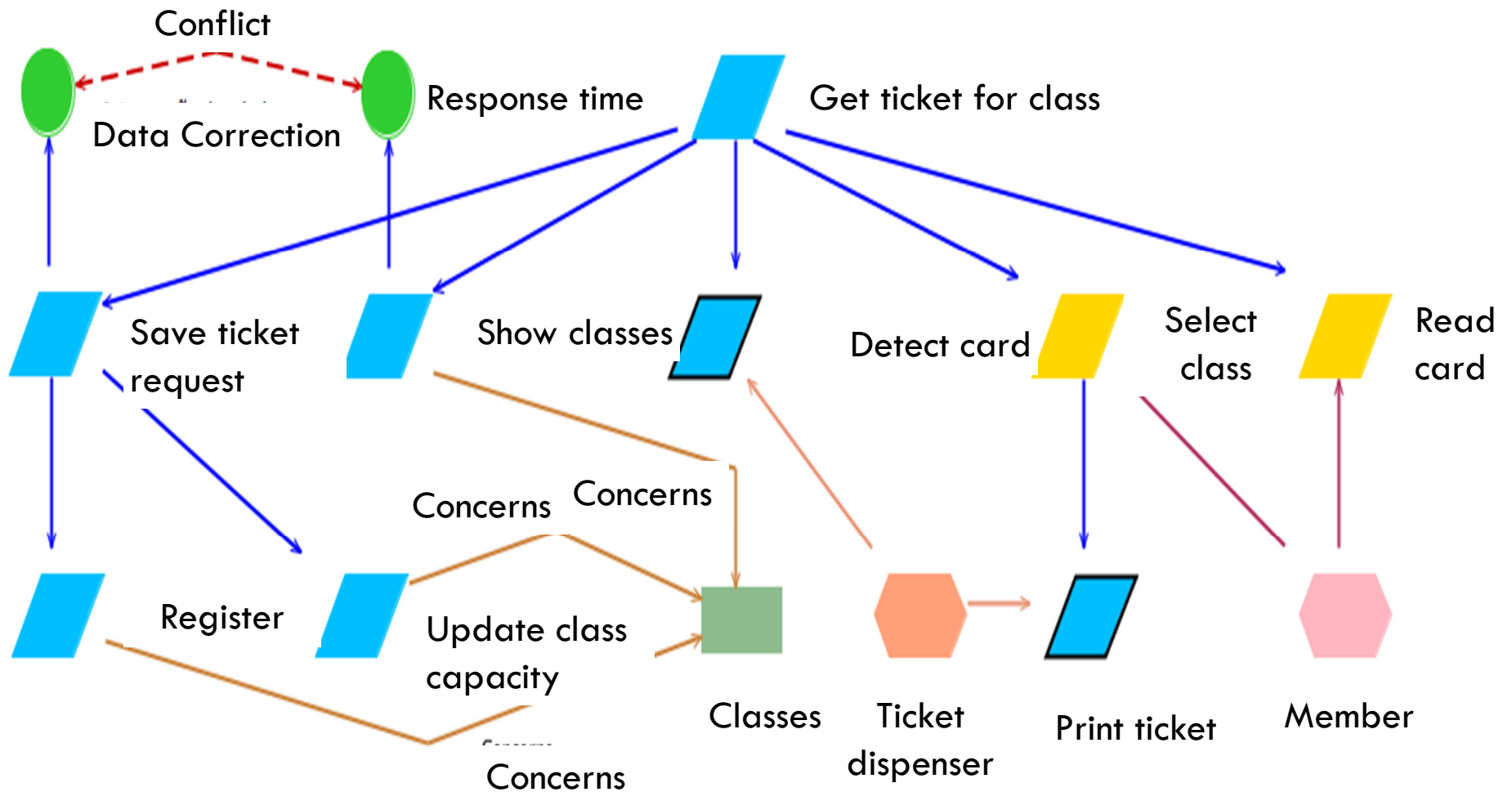
5

- ❑ The members of a health club need to get a ticket before participating in a specific class...



Let us work out... in KAOS

6



Add a few more requirements to our example

7

- 5 main functionalities
- 15 agents
- 212 sub-goals

- Is this model **complete**?
- How **complex** is this model?
- Is this complexity really necessary?



- **GORE aimed at large scaled systems**
- **Models can become really hard to understand**

Research objectives

8

- Analyse the extent to which a model is close to being **complete**
- Assess model **complexity** to identify model refactoring opportunities, e.g.:
 - Models may have a too deep goal hierarchy
 - Agents may have too many responsibilities
- **Prevent unanticipated extra costs** in the development phase
 - Better management of the completeness and complexity of the models

Contributions

- ❑ Tool supported approach in the metrics-based evaluation of the completeness and complexity of KAOS goal models
- ❑ The developer can measure the current status of his model and take on corrective actions, during model construction.
- ❑ The tool support is based on the integration of a KAOS editor with a KAOS metrics suite and
 - ▣ targeted to the requirements elicitation process,
 - ▣ it can also support *post-mortem* analysis from which lessons can be learned for future projects.

Contributions(2)

- Metrics and formally define all metrics using OCL
- We validate the metrics set and their implementation by extending an existing tool for editing KAOS goal models
 - ▣ modularKAOS developed in MDD on top of Eclipse

Approach outline

11

- Metrics **identification** using the Goal-Question-Metric approach
- Metrics (semi-)formal **definition** using OCL
- Metrics **evaluation** with real-world case studies
 - ▣ Often used as example of best practices using KAOS
- KAOS models analysis with metrics support

Goal: KAOS models completeness evaluation

12

Question	Metric
Q1. How close are we to completing the assignment of all goal responsibilities to agents?	PLGWA. Percentage of Leaf Goals With an Agent .
Q2. How detailed is the goal model with respect to objects?	PLGWO. Percentage of Leaf Goals With an Object .
Q3. How close are we to complete the resolution of all the goal obstacles?	PLOWS. Percentage of Leaf Obstacles With a reSolution .
Q4. How detailed is the goal model with respect to operations?	PLGWOp. Percentage of Leaf Goals With an Operation .
Q5. How well supported are the operations in the goal model?	POpWA. Percentage of Operations With an Agent .

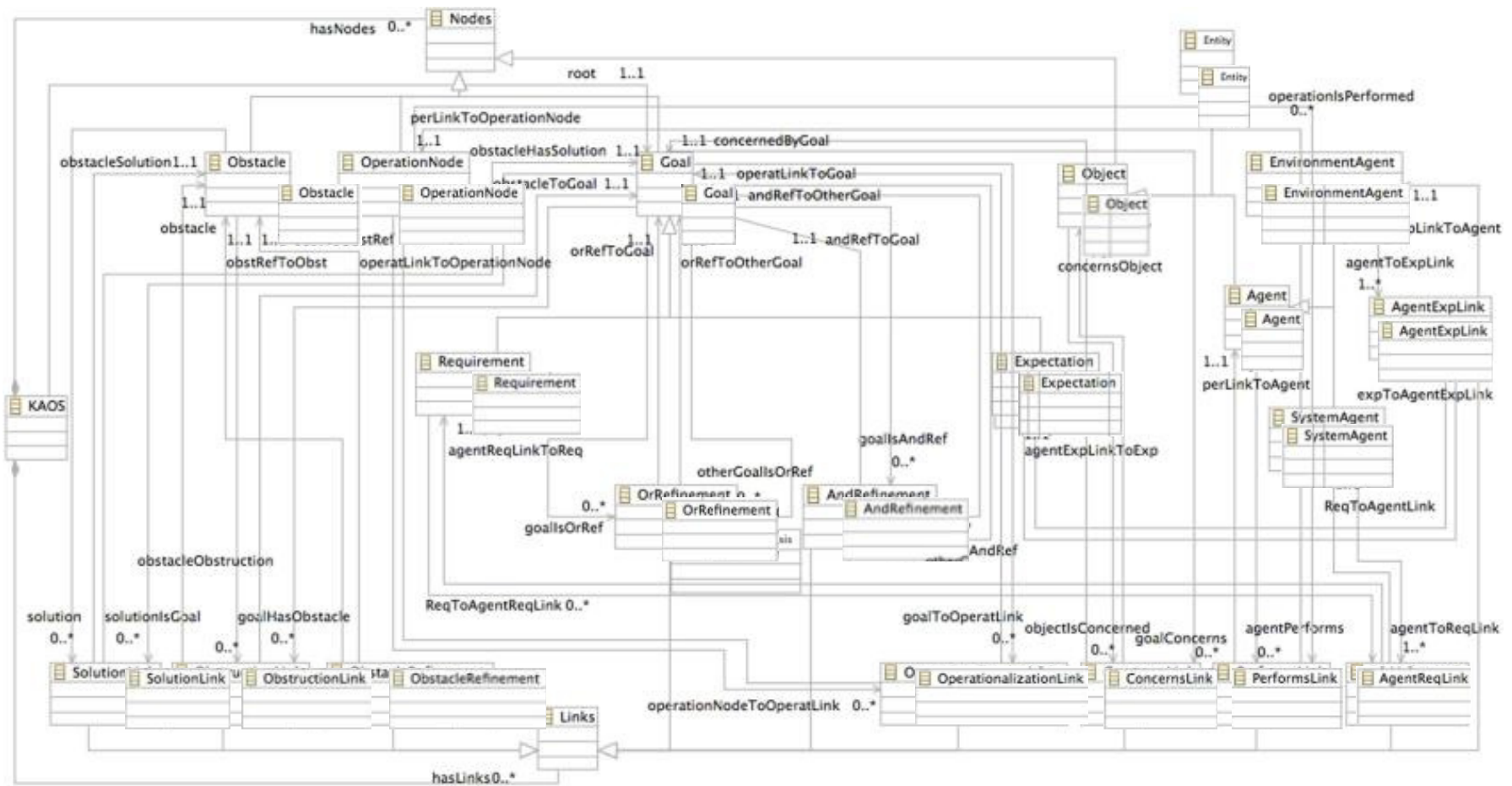
Goal: KAOS models complexity evaluation

13

Question	Metric
Q6. Does an agent have too much responsibility in the model?	ANLG. Number of Leaf Goals per Agent.
Q7. Does a leaf goal have too many/few objects?	GNO. Number of Objects per Goal.
Q8. How difficult is it to understand a model, with respect to the number of refinement levels?	MD. Model Depth.
Q9. How complex is a model, with respect to its goal refinements?	RNSG. Root Number of Sub-Goals.

modularKAOS: partial metamodel

14

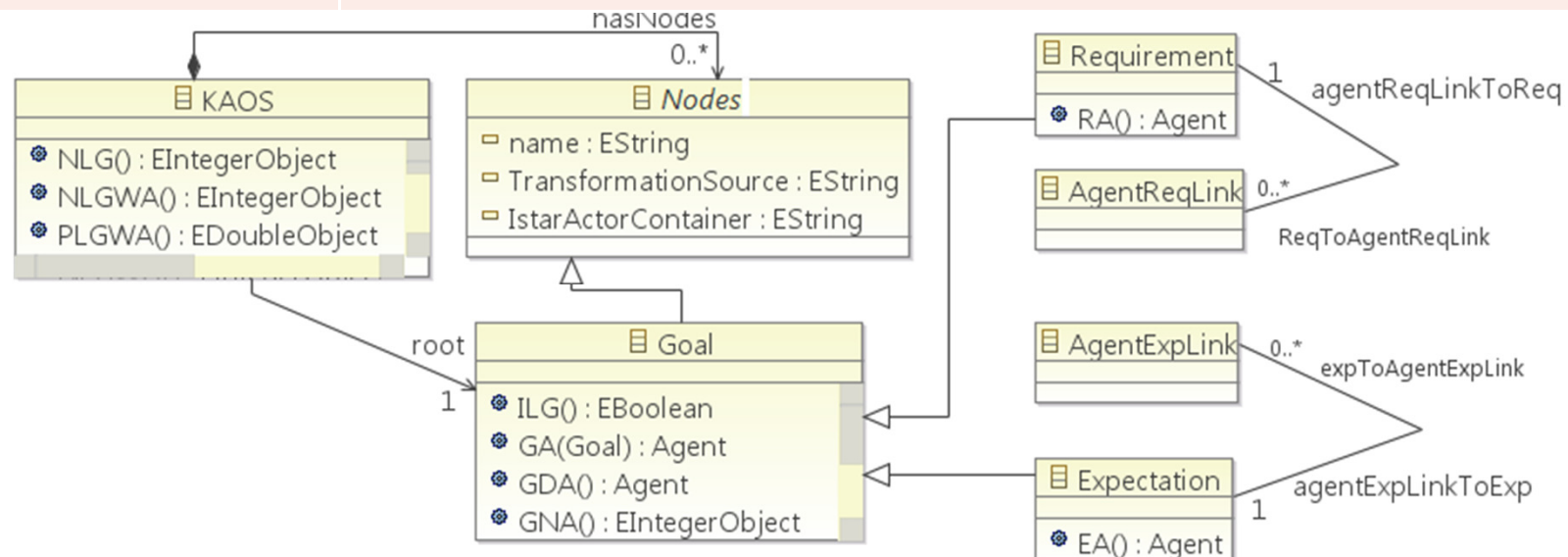


Metrics definition

Q1 - How close are we to completing the assignment of all goal responsibilities to agents?

15

Name	PLGWA – Percentage of Leaf Goals With an Agent
Informal definition	Percentage of leaf goals that have an associated agent in the model.
Formal definition	context KAOS def: <u>PLGWA(): Real</u> = self.NLGWA() / self.NLG()
Pre-condition	context KAOS::PLGWA() pre: self.NLG() > 0
Comments	If there are no leaf goals the result is undefined. This requires: NLG – Number of Leaf Goals NLGWA– Number of Leaf Goals With an Agent
Recommendation	In a complete model, all leaf goals should be assigned to an agent.

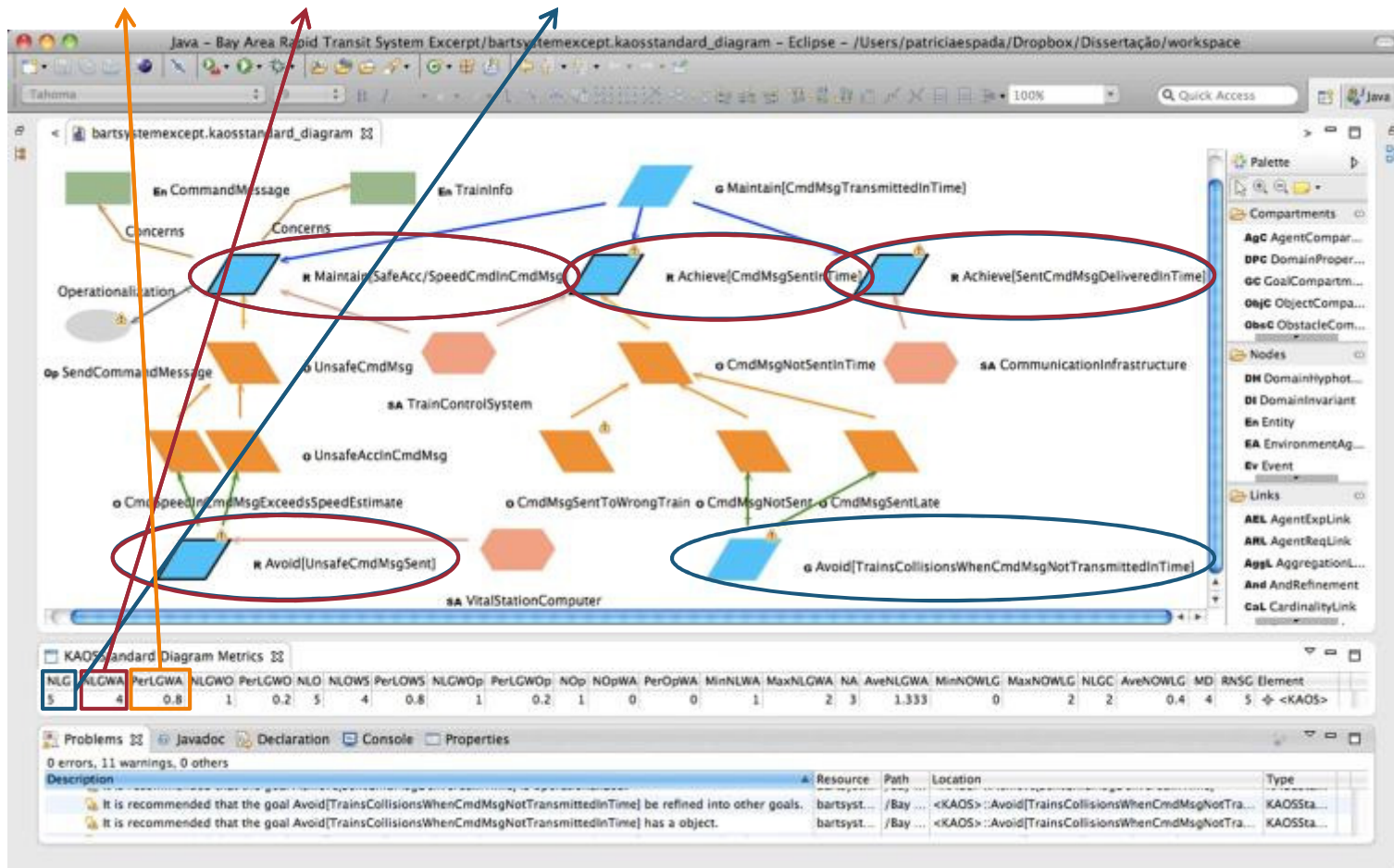


Computing % of leaf goals with an agent

16

$$\text{PLGWA} = \text{NLGWA} / \text{NLG}$$

$$\text{PLGWA} = 4 / 5 = 0.8$$



17

Evaluation



BARTS



MSCS



ES



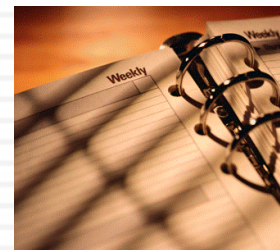
CPMS



LMS



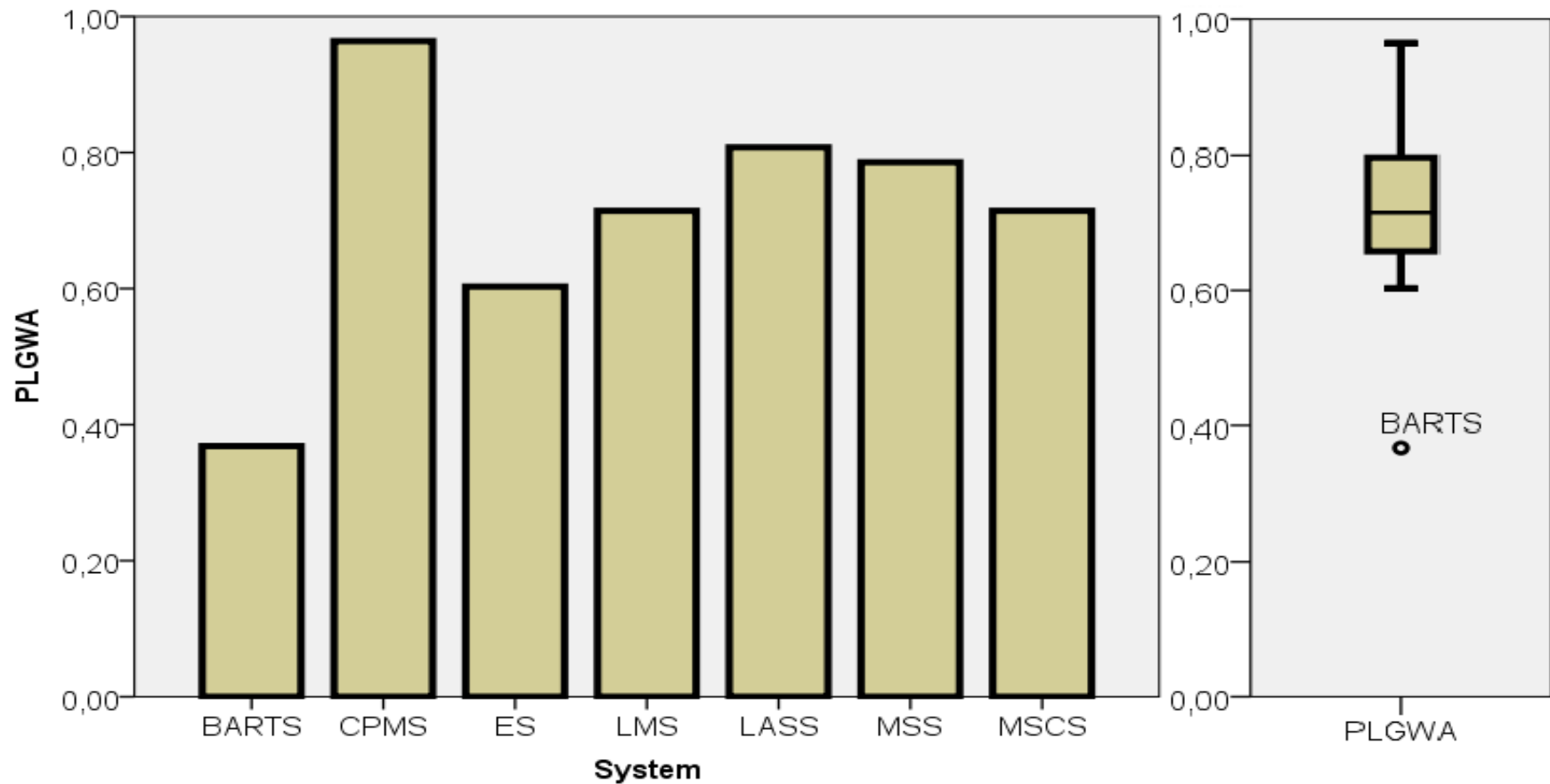
LAS



MS

Percentage of Leaf Goals with an Agent

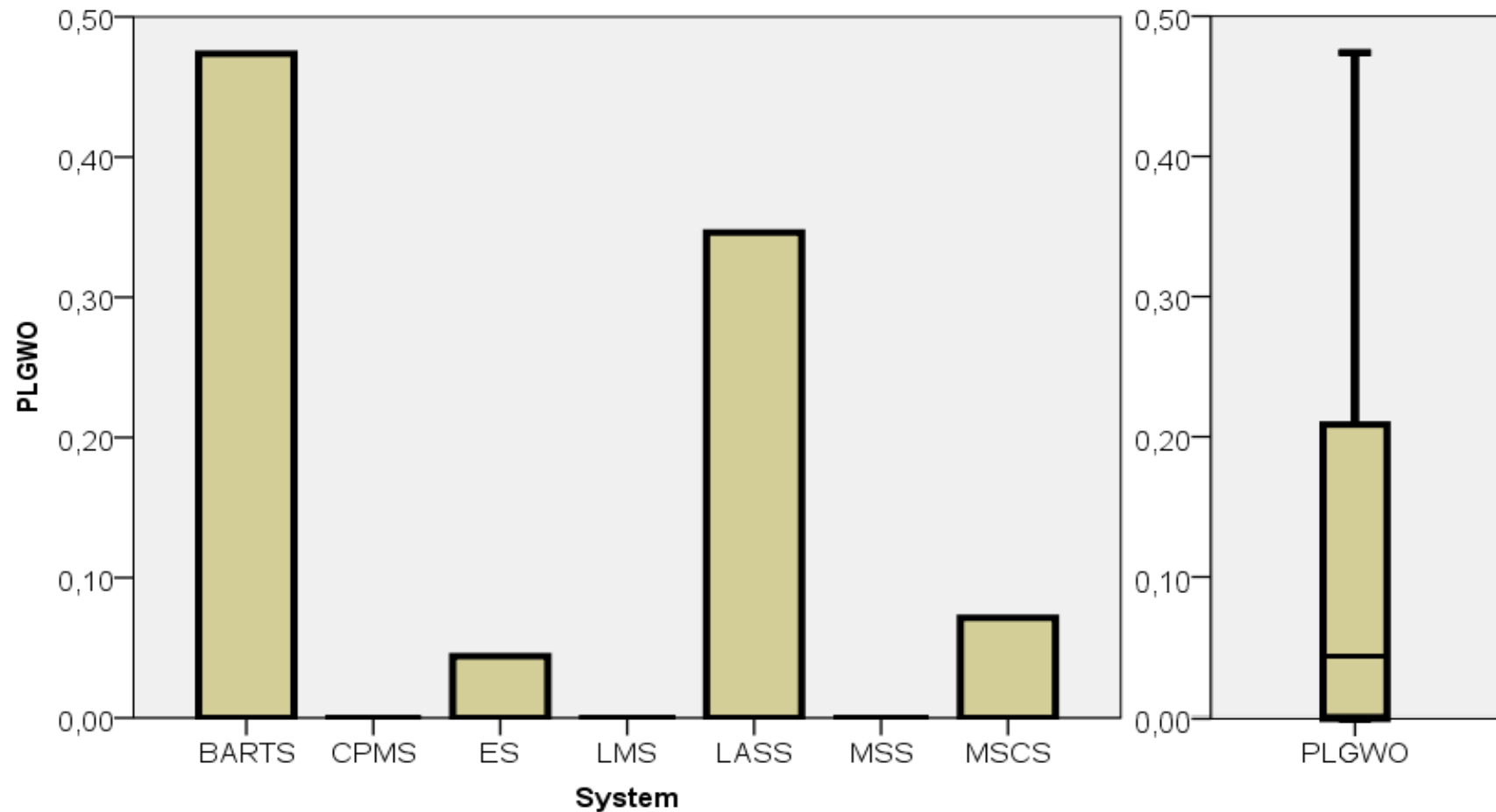
18



Percentage of Leaf Goals with an Object

19

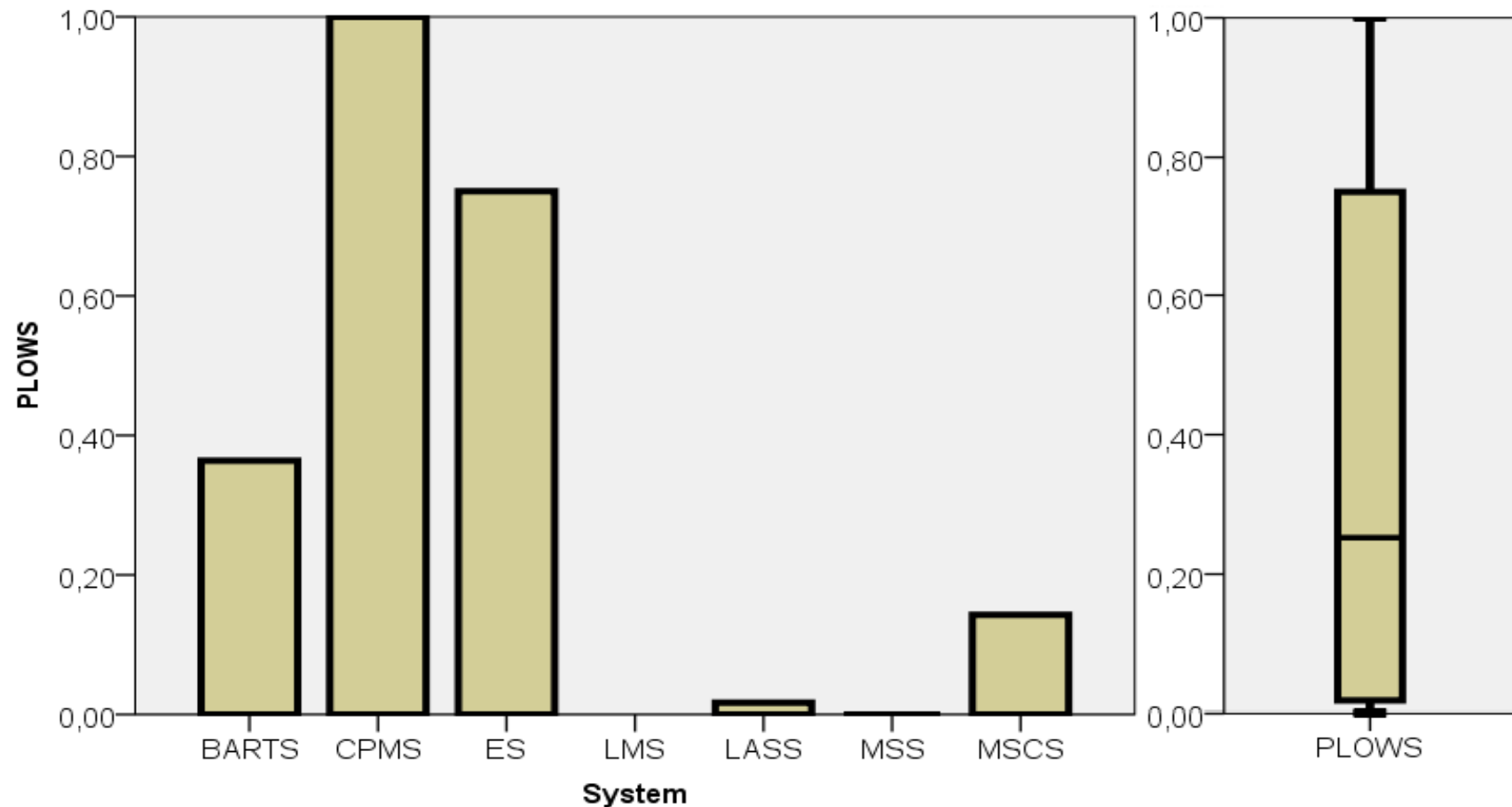
Espada, Goulão, Araújo, "A Framework to Evaluate Complexity and Completeness of KAOS Goal Models", CAiSE 2013, Valencia, Spain



Percentage of Leaf Obstacles with a reSolution

20

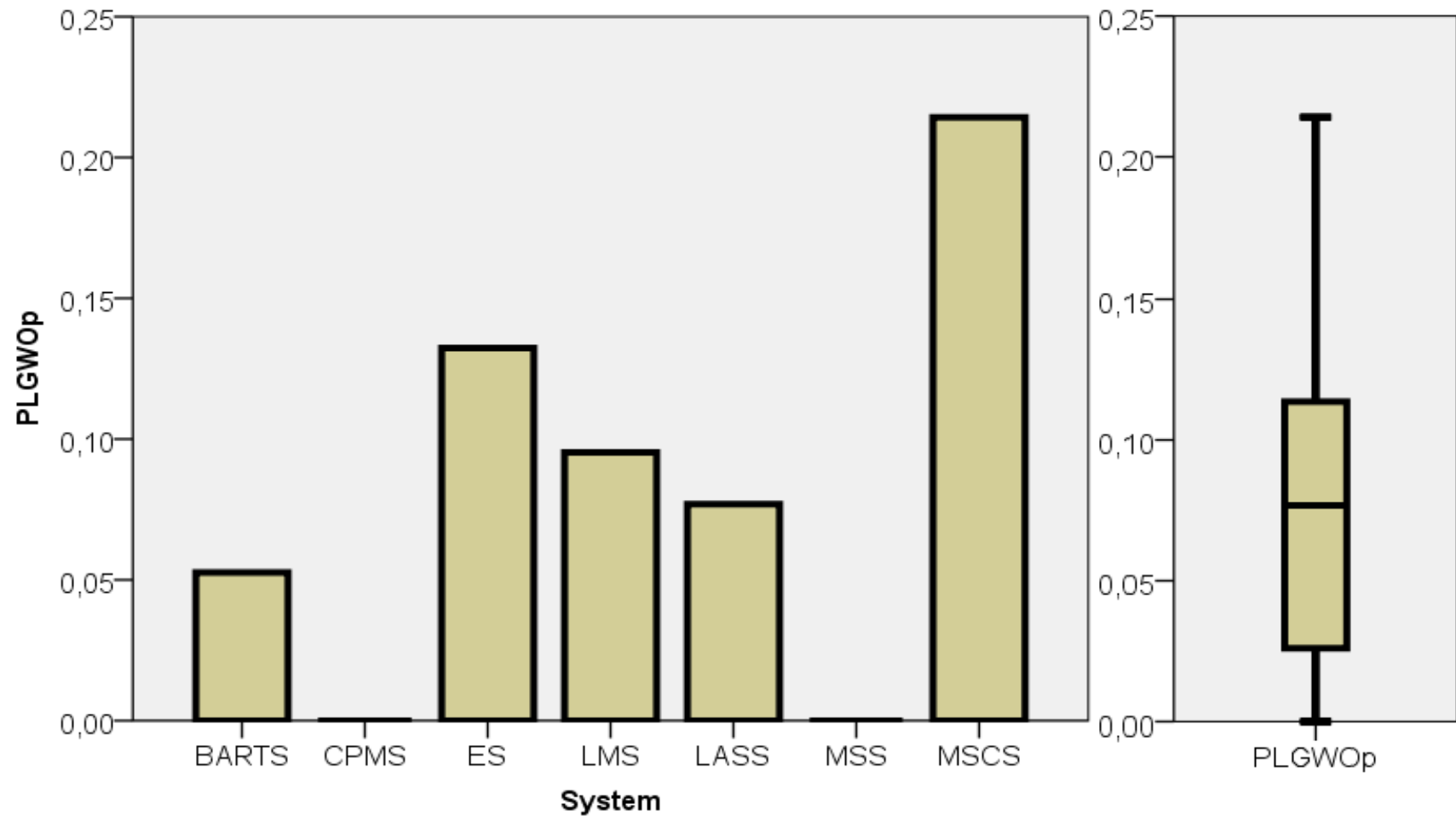
Espada, Goulão, Araújo, "A Framework to Evaluate Complexity and Completeness of KAOS Goal Models", CAiSE 2013, Valencia, Spain



Percentage of Leaf Goals with an Operation

21

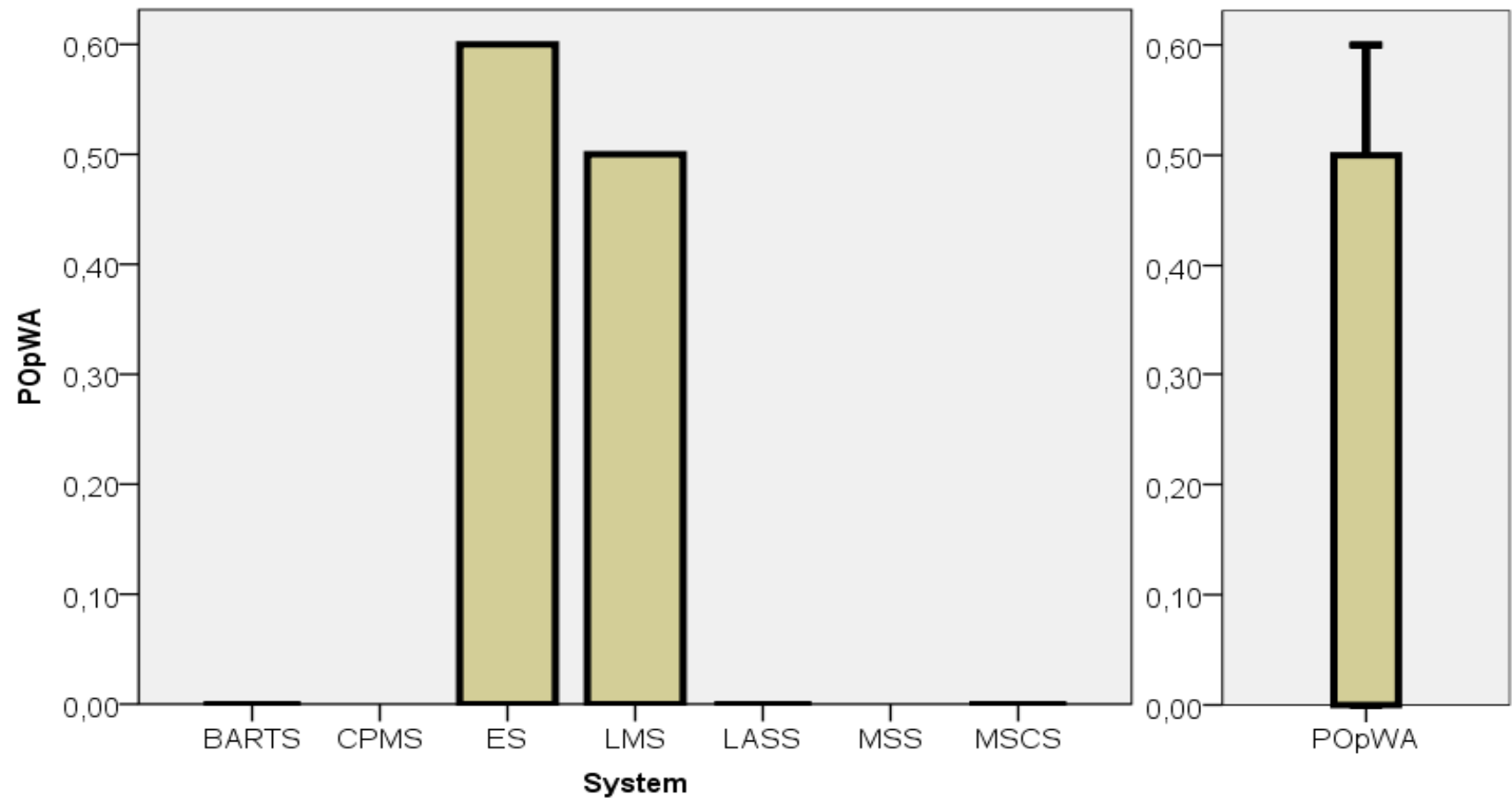
Espada, Goulão, Araújo, "A Framework to Evaluate Complexity and Completeness of KAOS Goal Models", CAiSE 2013, Valencia, Spain



Percentage of Operations with an Agent

22

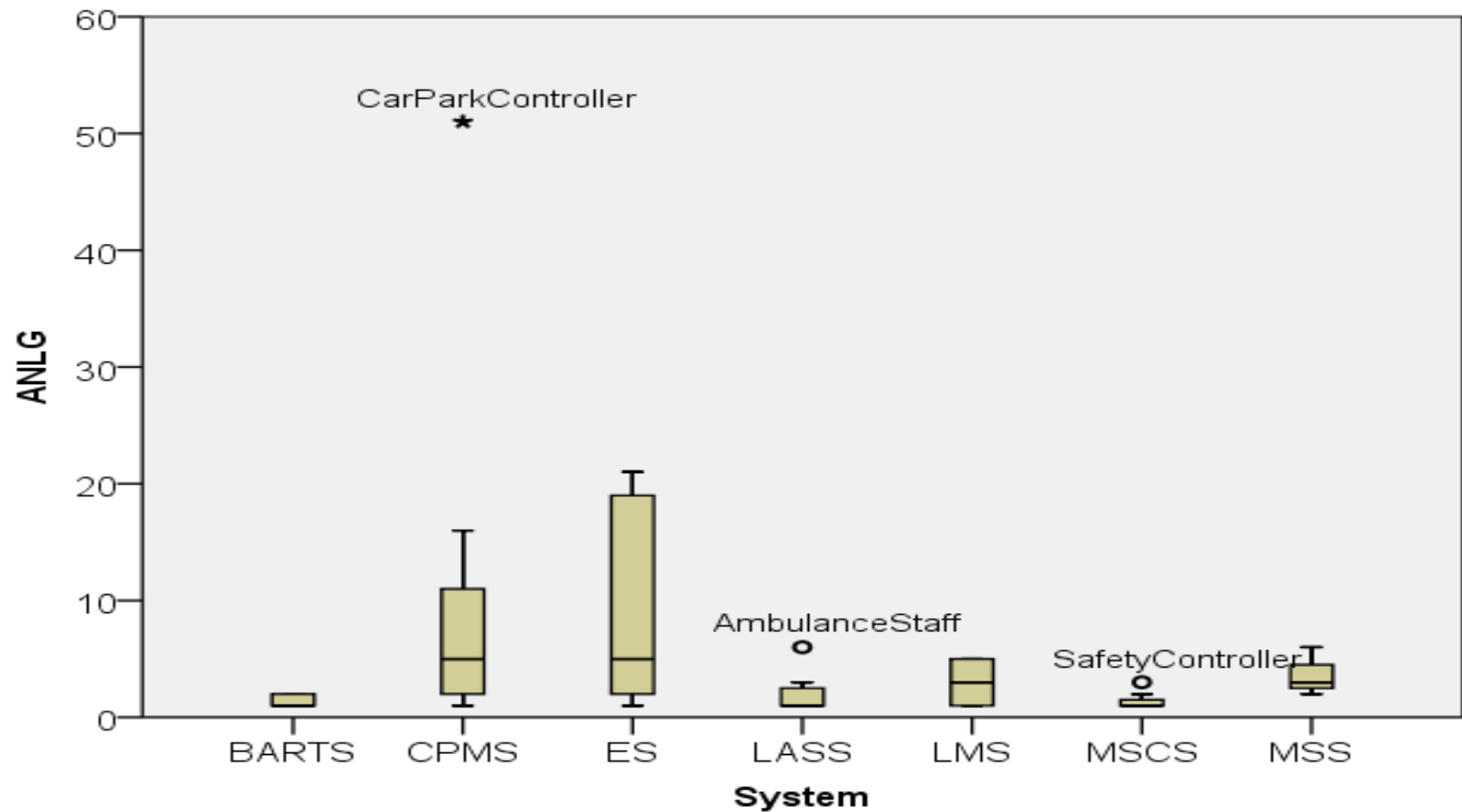
Espada, Goulão, Araújo, "A Framework to Evaluate Complexity and Completeness of KAOS Goal Models", CAiSE 2013, Valencia, Spain



Number of Leaf Goals per Agent

23

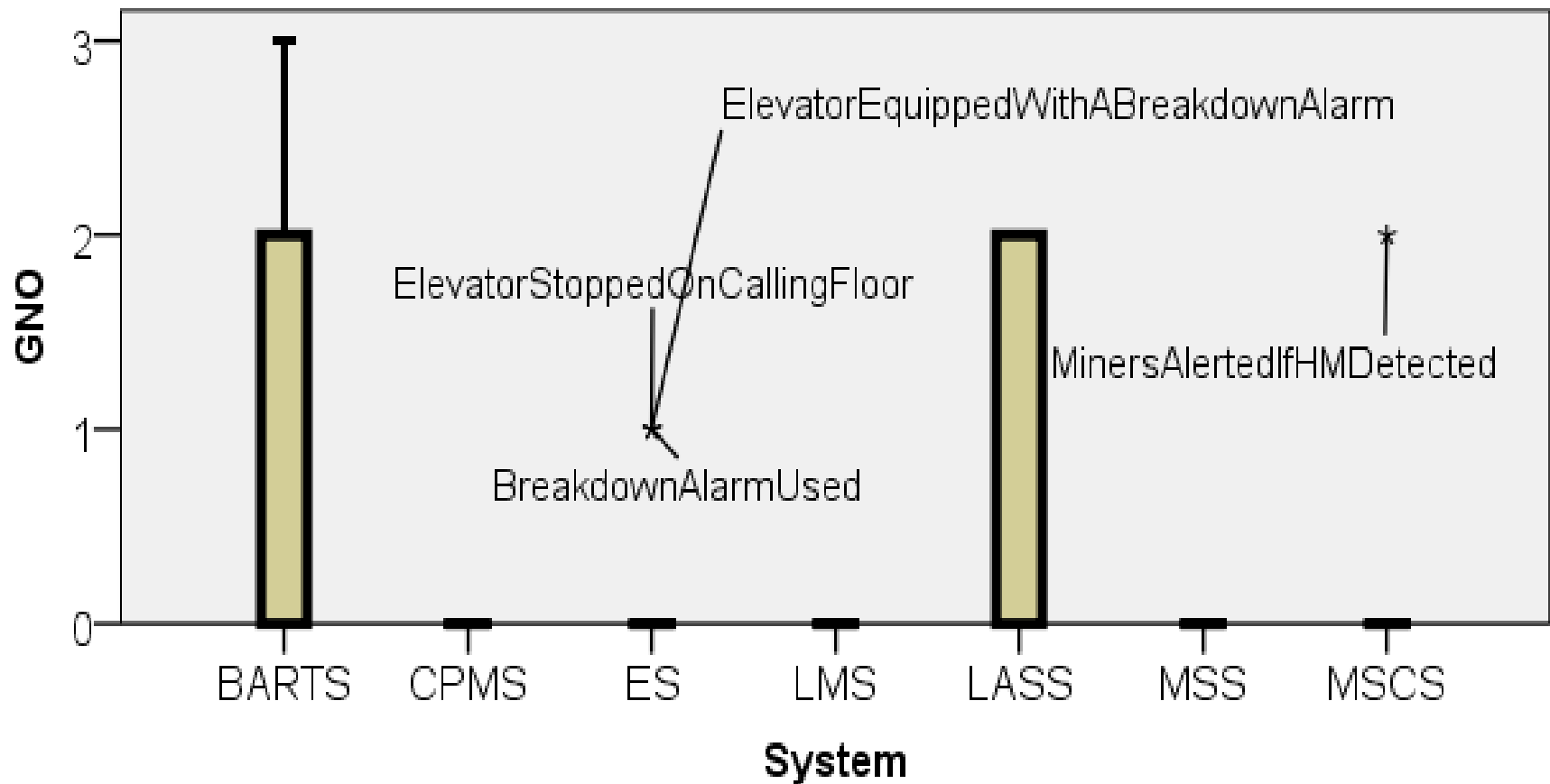
Espada, Goulão, Araújo, "A Framework to Evaluate Complexity and Completeness of KAOS Goal Models", CAiSE 2013, Valencia, Spain



Objects per Goal

24

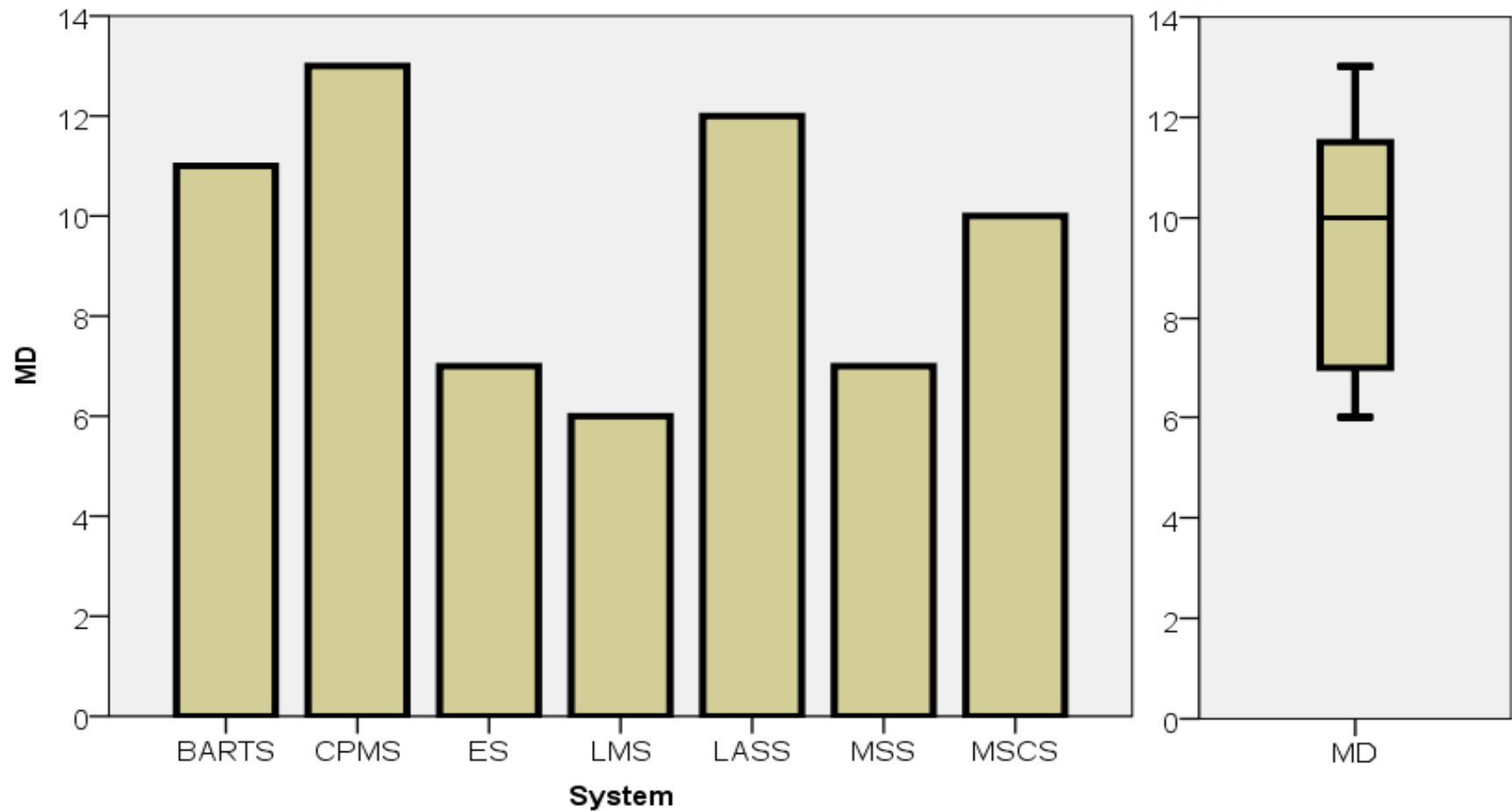
Espada, Goulão, Araújo, "A Framework to Evaluate Complexity and Completeness of KAOS Goal Models", CAiSE 2013, Valencia, Spain



Model Depth

25

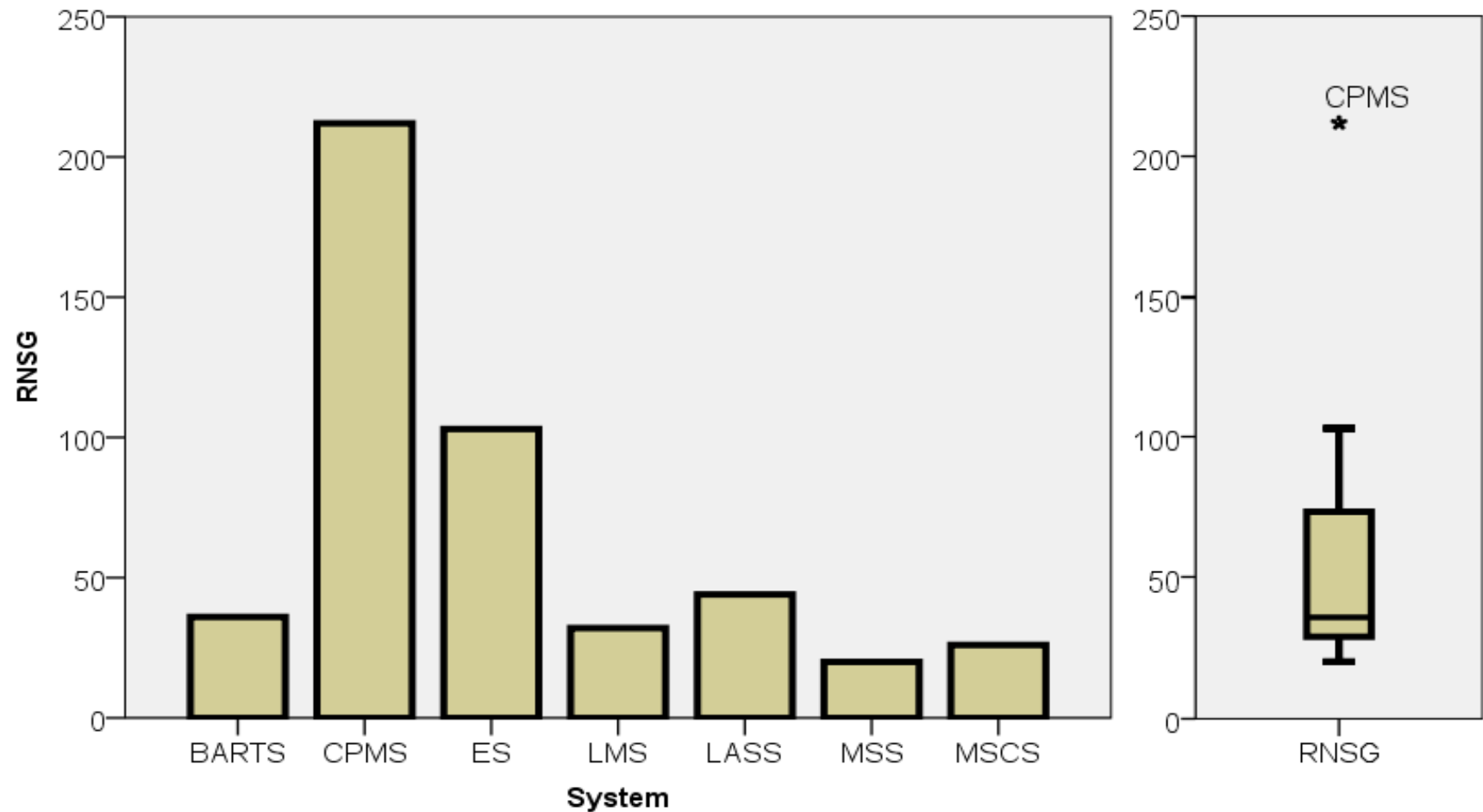
Espada, Goulão, Araújo, "A Framework to Evaluate Complexity and Completeness of KAOS Goal Models", CAiSE 2013, Valencia, Spain



Root Number of Sub-Goals

26

Espada, Goulão, Araújo, "A Framework to Evaluate Complexity and Completeness of KAOS Goal Models", CAiSE 2013, Valencia, Spain



Discussion (Completeness)

27

- ❑ Most models handle responsibility assignment of leaf goals to agents
- ❑ Objects are not frequently used
- ❑ When obstacles are specified, we find a big variation (from 0% to 100%) of the percentage of obstacles with a resolution
- ❑ Operations are even more rarely used than objects
- ❑ Only two of the case studies model the assignment of operations to agents, showing this is a fairly unexplored modeling feature.

Discussion (complexity)

28

- Relatively few leaf goals assigned to an agent
 - ▣ Do not attribute too many responsibilities to a single agent
- Assigning objects to goals is a mostly unexplored feature of models
- Model depth varies much less than the number of model elements, suggesting a fairly consistent state of practice with respect to what is considered an adequate model decomposition level
- Big variations in the case studies, concerning the number of subgoals defined in each model
 - ▣ The average number is around 40 subgoals, although in one of the examples it is over 200 goals.

Limitations

- Framework does not cover
 - ▣ **Quality** of elicited requirements
 - ▣ **Thoroughness** of elicited requirements
- No easy essential vs. **accidental** complexity differentiation
 - ▣ No reference values for what is “acceptable”
- Only “good” examples in the sample
 - ▣ Reference for best practices, but not for bad ones

Conclusions

- ❑ Metrics suite for **completeness** and **complexity** of KAOS goal models
- ❑ Values computed and updated as the model evolves
- ❑ Full integration with modeling tool
 - ▣ Completeness monitoring to help assessing effort to model completion
 - ▣ Complexity monitoring to detect early potential quality problems and identifying refactoring opportunities
- ❑ Proof of concept with best practices examples of KAOS models
 - ▣ Obtained metrics values are a step towards deeper understanding of actual goal modeling practices

Future work

31

- Metrics set extension to quality attributes
- Evaluation replication with other KAOS models
 - ▣ Towards metrics-based modeling heuristics
- Assess completeness in terms of requirements coverage
 - ▣ Trace model elements to requirements sources
 - ▣ Identify the requirements in those sources that are yet to be covered by the goal models

2nd Part: Creatively Building KAOS Models

In collaboration with Fernando Wanderley
(presented at MoDRE'13, workshop of RE'13)

Goals

- Provide a systematic and modular goal-oriented modelling process...
- Helping the requirements engineers with elicitation of KAOS concerns (e.g. agents, goals and objects) and...
- Generate (or represent) KAOS models from mind maps, through model driven techniques, providing better understanding and communication by stakeholders (e.g. domain expert and business specialist)



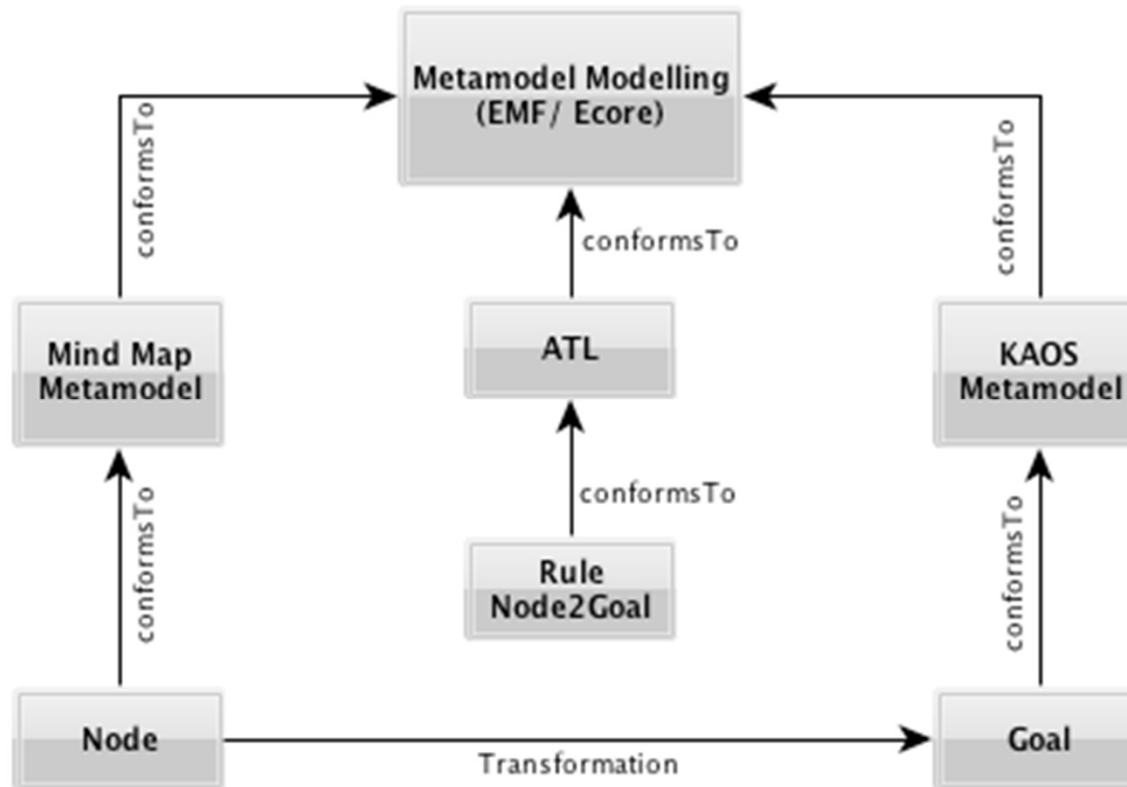
Mind Map



Mind Mapping Modelling Adoptions

- Both the academy and the industry consider several techniques.
 - ▣ Tools based on mind maps powerful tools for managing the process of eliciting requirements in agile development
- We have been investigating how mind maps can be used in requirements engineering to facilitate communication among the domain experts and...
- How model driven engineering techniques can help generating requirements models from mind maps;

Model-Driven Engineering



Mind Map Metamodel

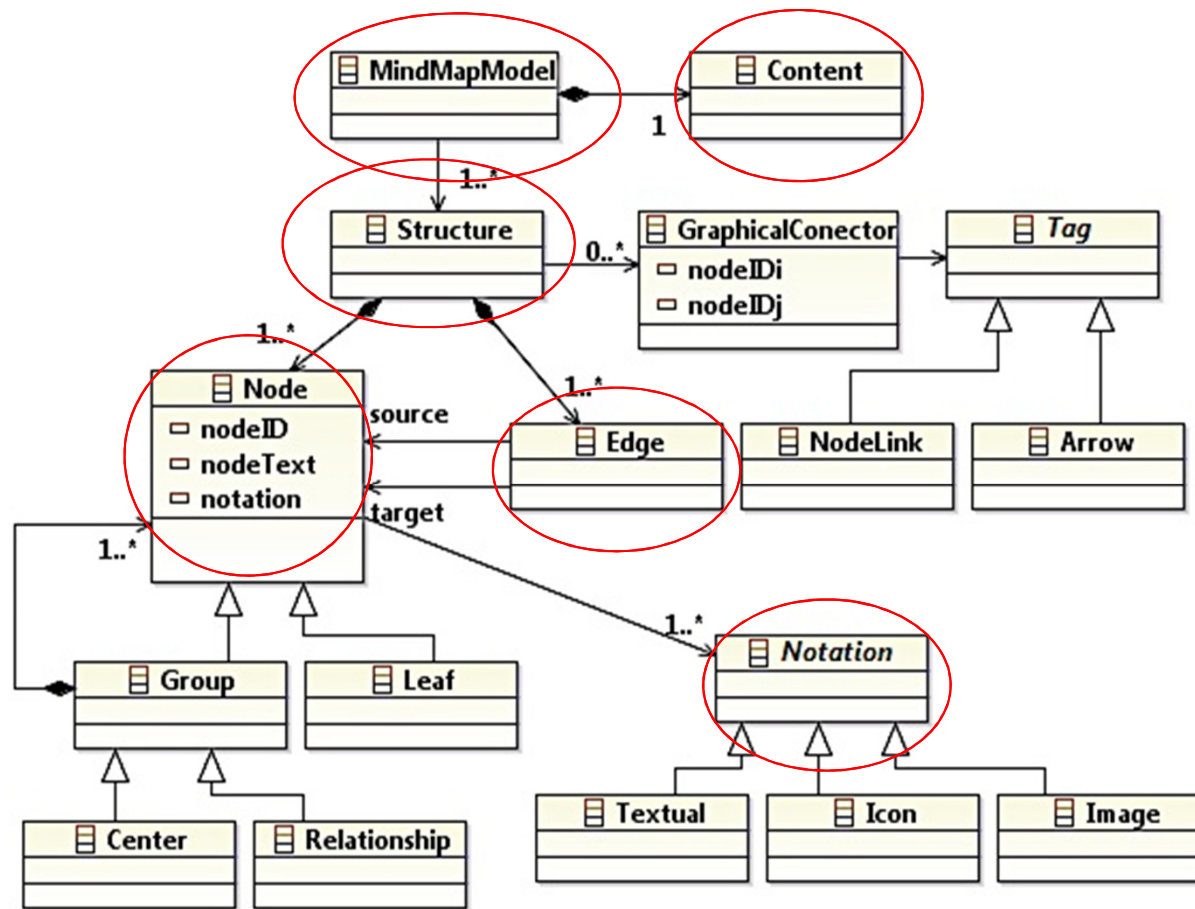
- The main objective of this article was to **establish the mapping between** the main elements and concepts of the KAOS and Mind Map models.

- Currently, mappings language

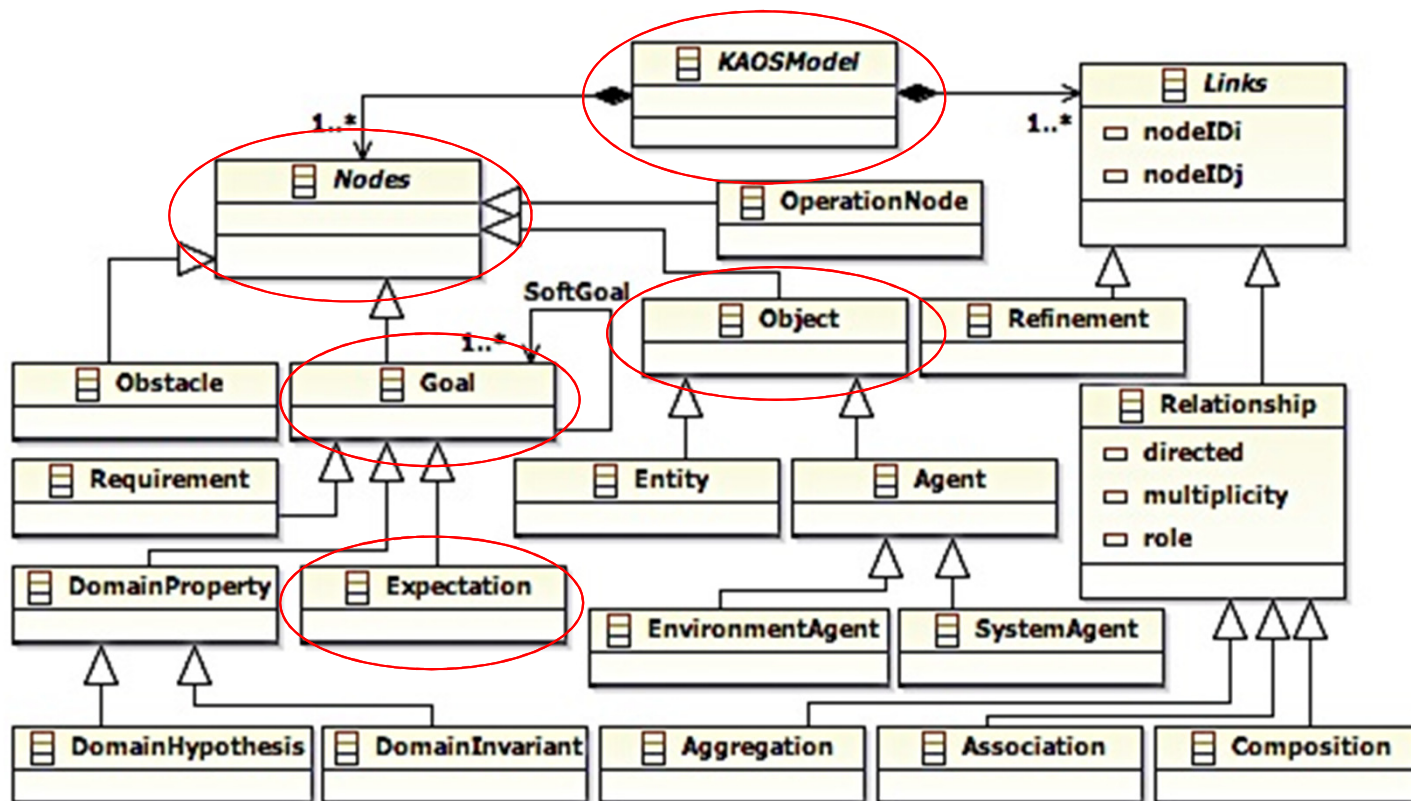


plementing these
TL transformation

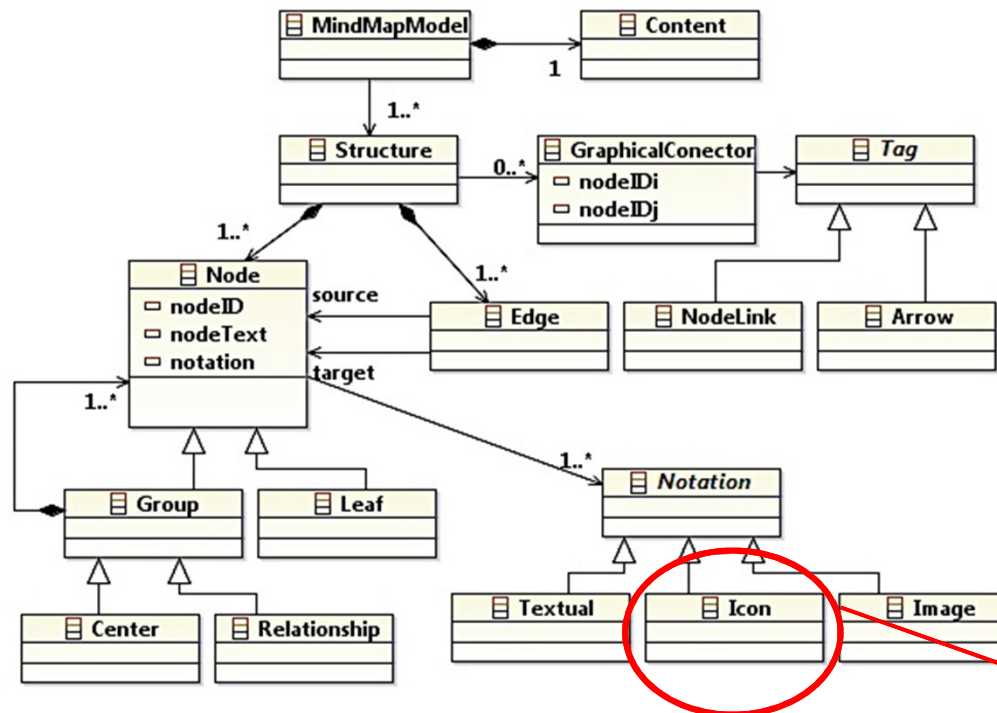
Mind Map Metamodel



KAOS Metamodel



Representation of Mapping Elements



KAOS Concerns

Element	Icon	Concern	Semantic
Requirement		Goal Model	<<edit>>
Expectation		Goal Model	<<idea>>
Domain Property		Goal Model	<<warning>>
Entity		Object Model	<<list>>

Set of Transformation Rules (by Concerns)

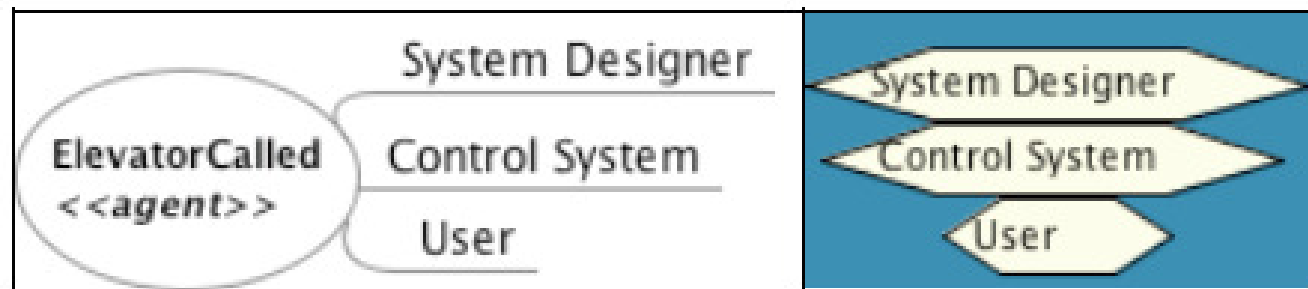
42

3rd International Model-Driven Requirements Engineering



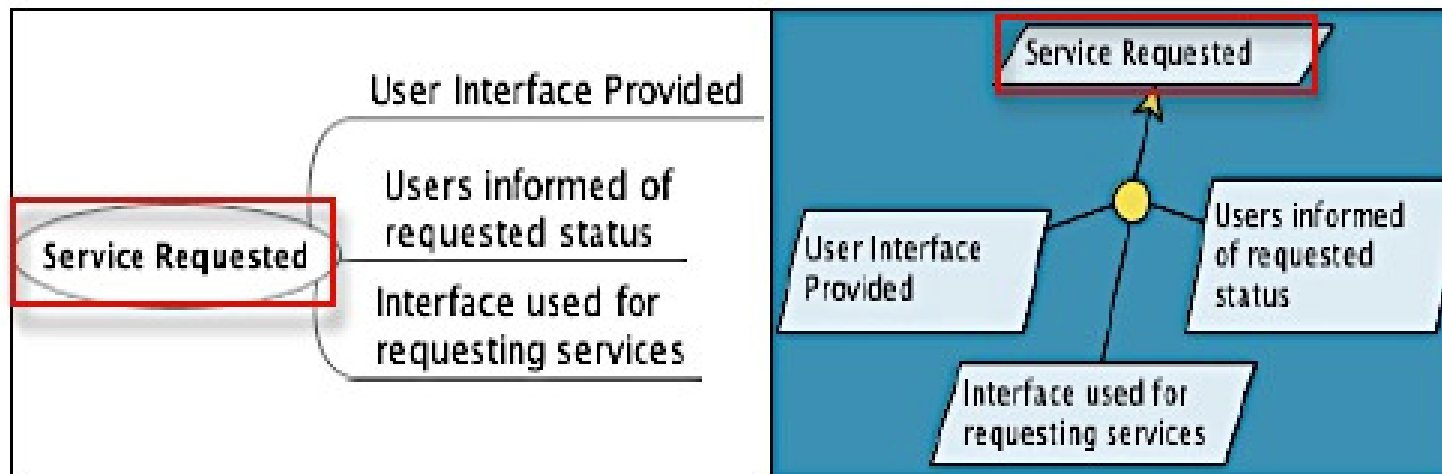
Agents Modelling

Rule 1. Each node different from the node root will be transformed into an Agent



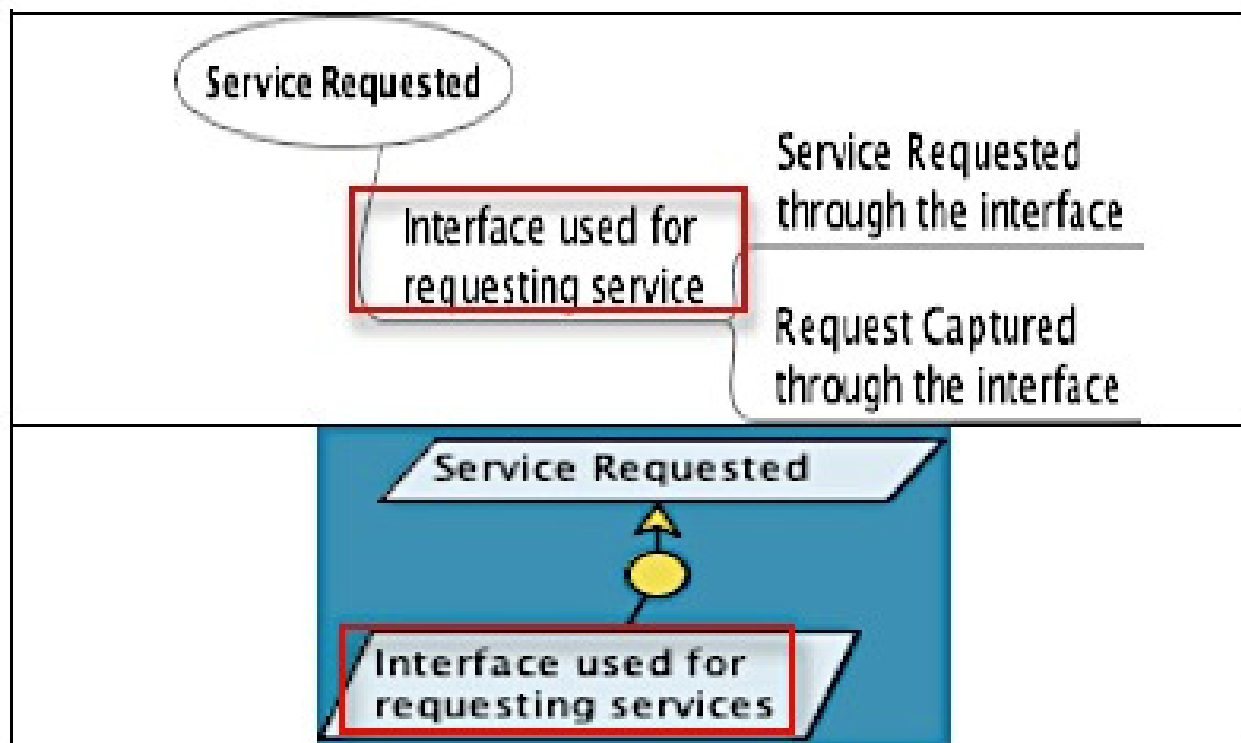
Goals Modelling

Rule 2. Each node root will be transformed into RootGoal



Goals Modelling

Rule 3. Each node, different from root node and not a leaf node, will be transformed into another Goal with a Refinement link from its parent node.



Goals Modelling

Rule 4. For each node different from root node and being a leaf node, we have three transformation possibilities depending on the node state:

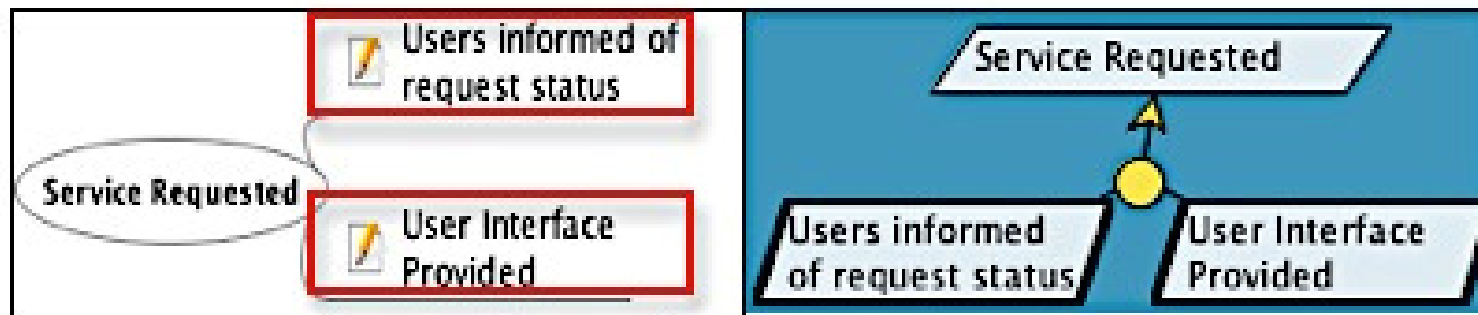
- 4.1 -Requirement
- 4.2 -Expectation
- 4.3 -DomainProperty

Goals Modelling

47

3rd International Model-Driven Requirements Engineering

Rule 4.1 Each node with requirement state  will be transformed into a Requirement

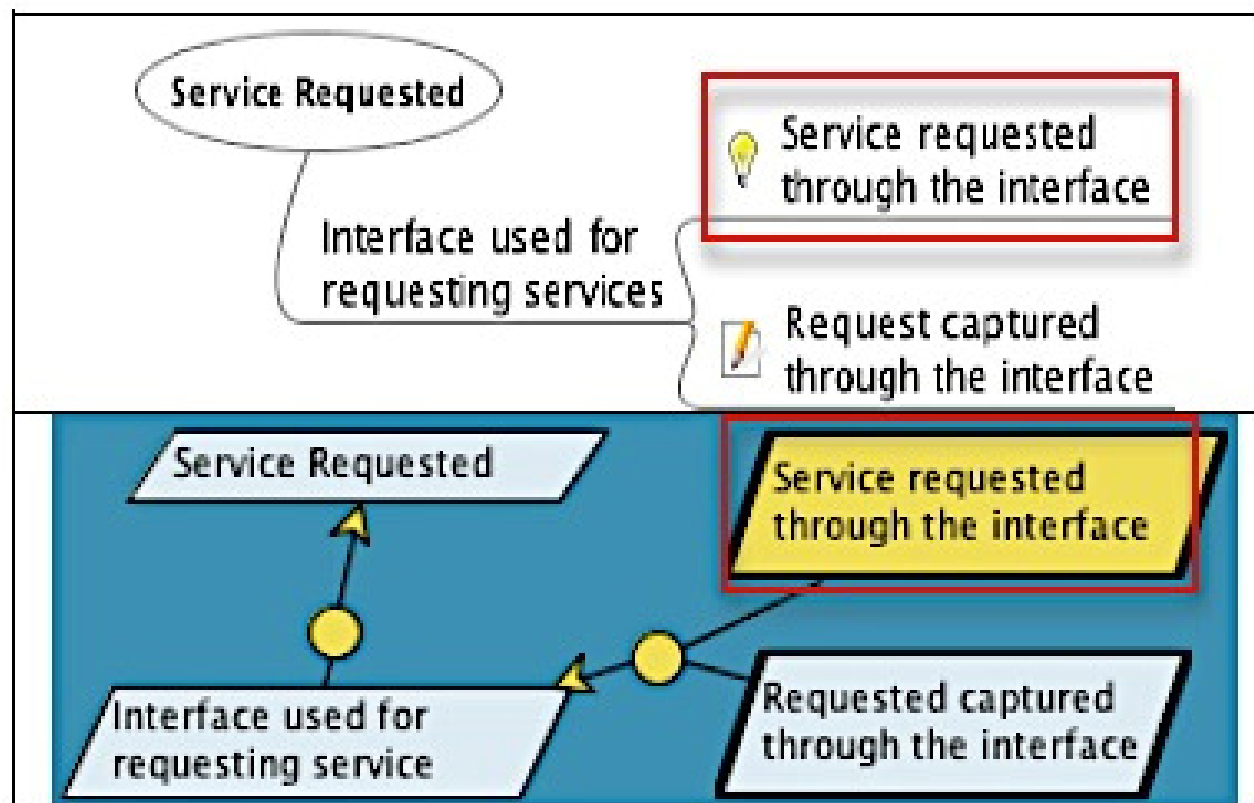


Goals Modelling

48

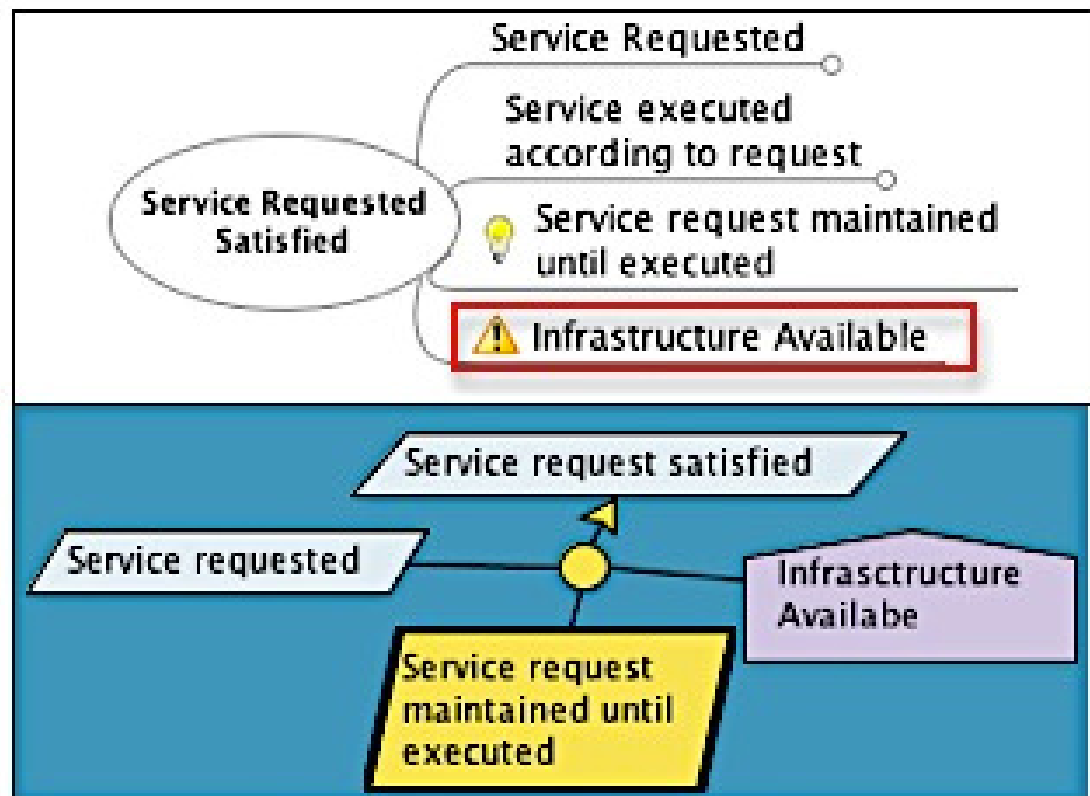
3rd International Model-Driven Requirements Engineering

Rule 4.2 Each node with “idea” state 💡 will be transformed into an Expectation.



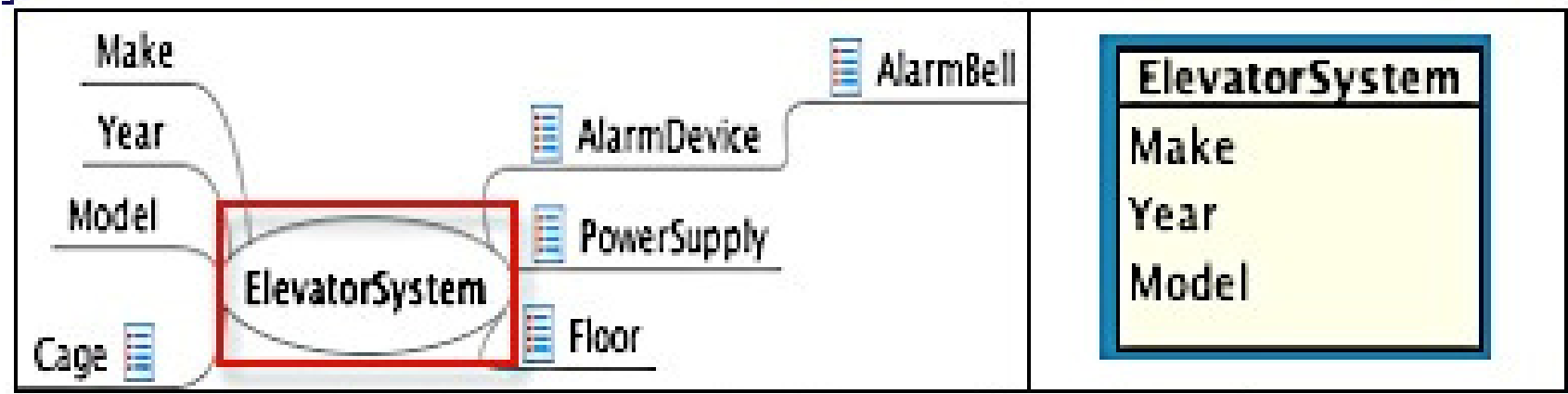
Goals Modelling

Rule 4.3 Each node with “warning” state ⚠ will be transformed into a DomainProperty.



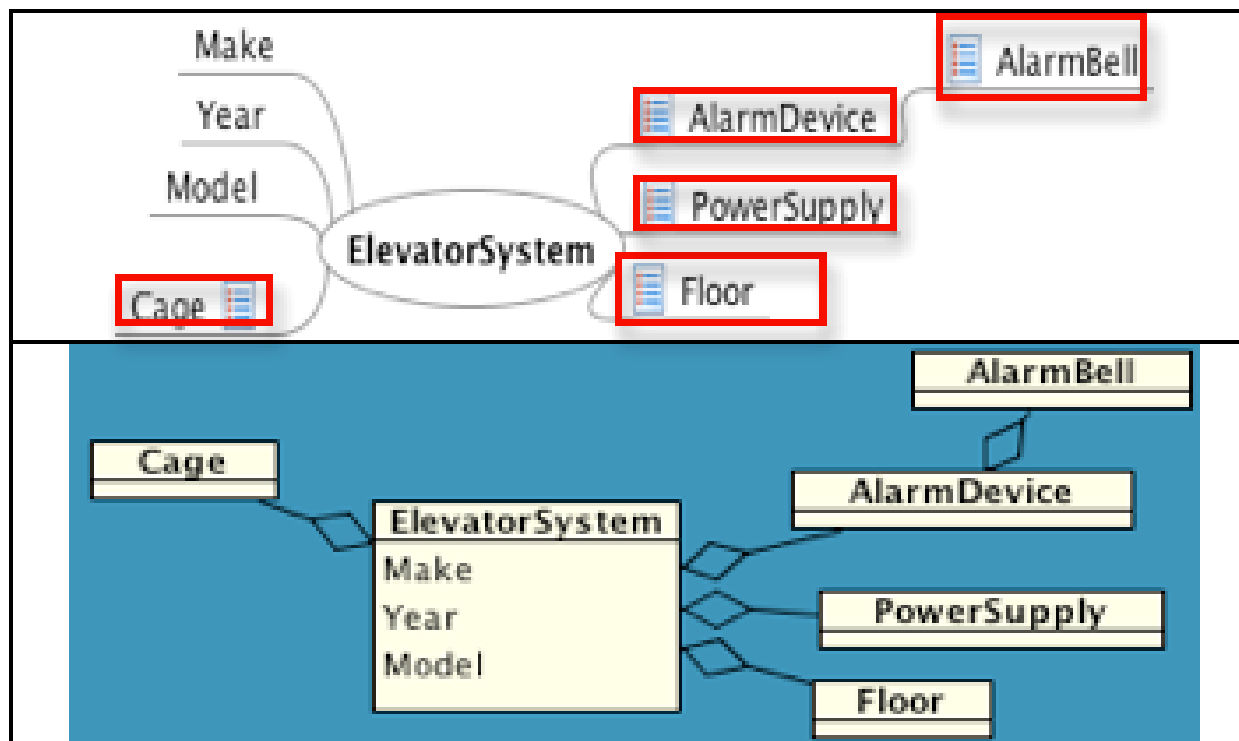
Objects Modelling

Rule 5. The root node of a mind map will always be transformed into a class.



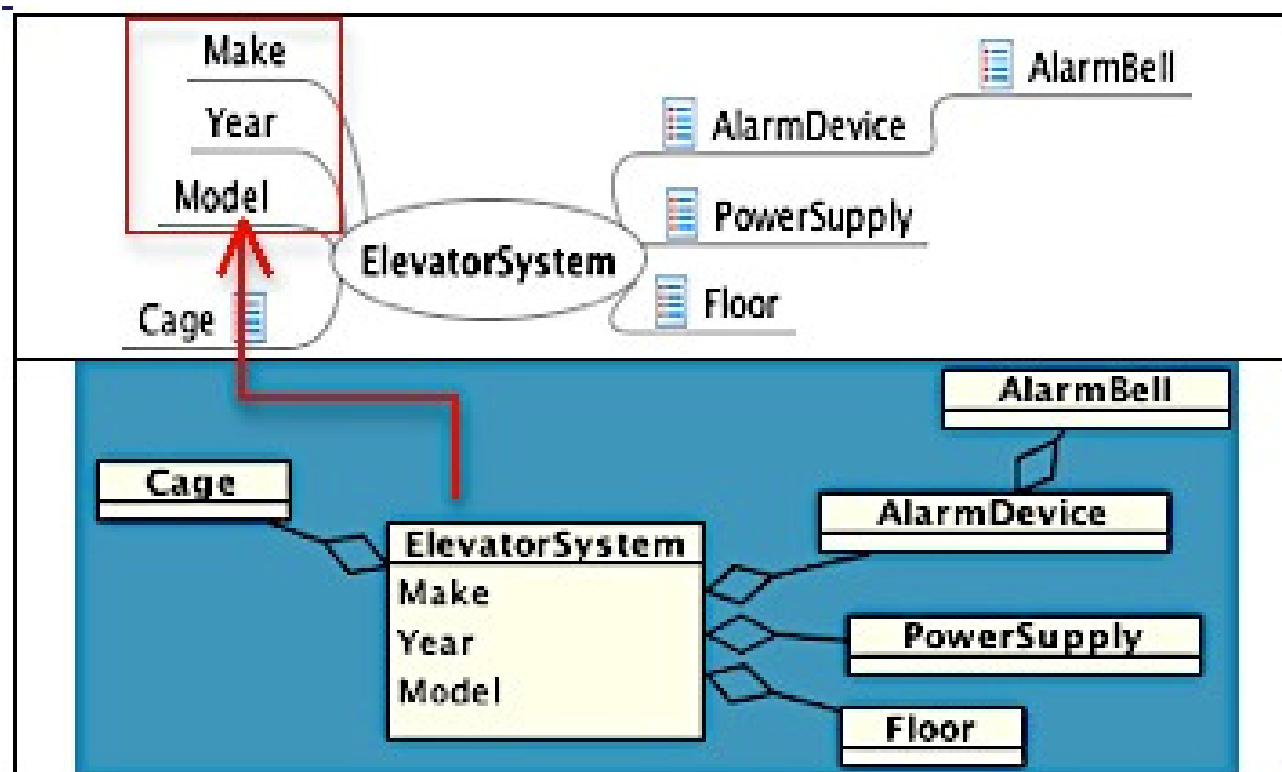
Objects Modelling

Rule 6. For each node, different from the root node and defined as an entity state ( this will be transformed into a class .

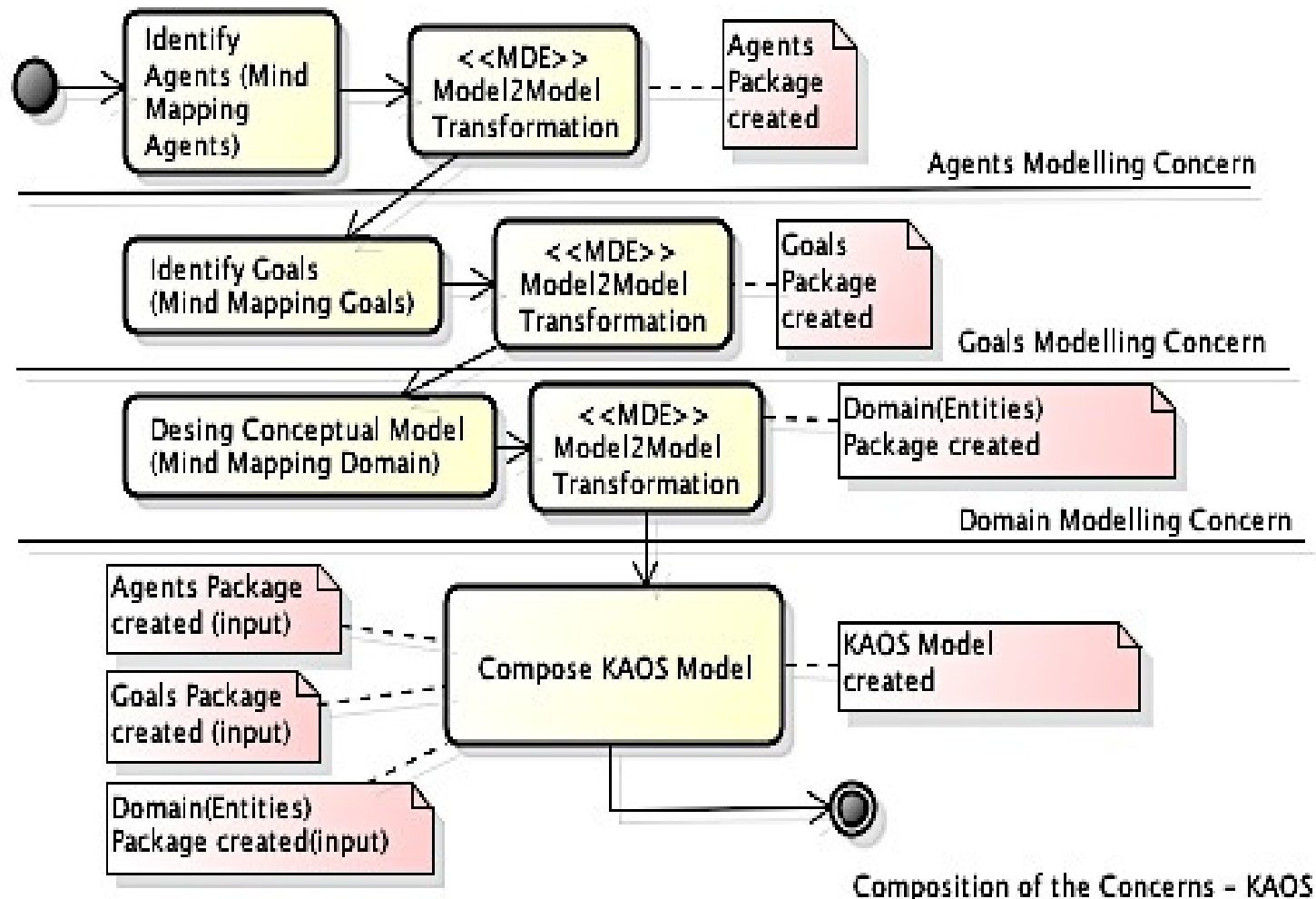


Objects Modelling

Rule 7. For each node, different from the root node, which is not an Entity, and child of node of type Entity - this will be transformed into an attribute.



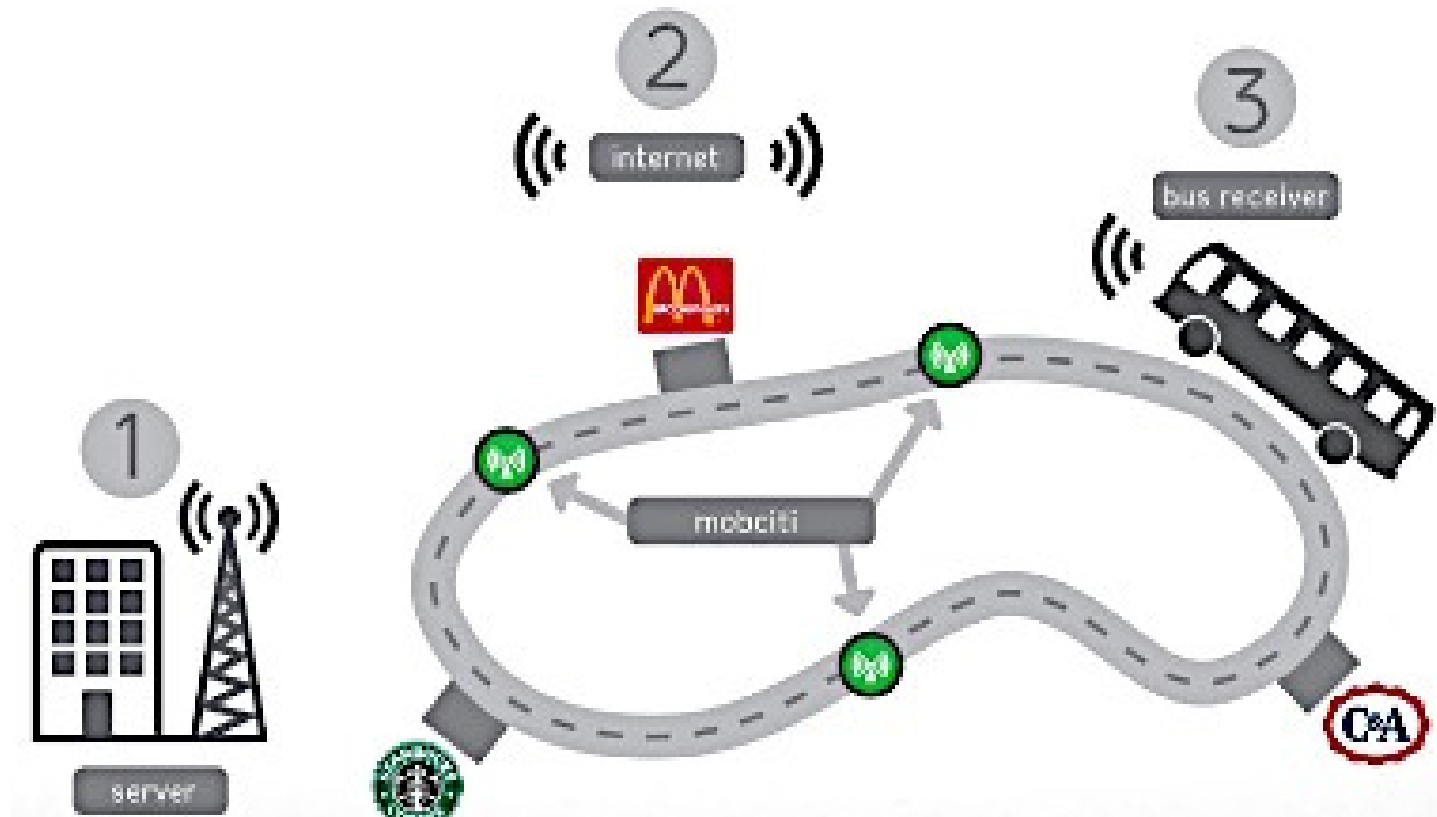
Systematic and Agile Transformation Process



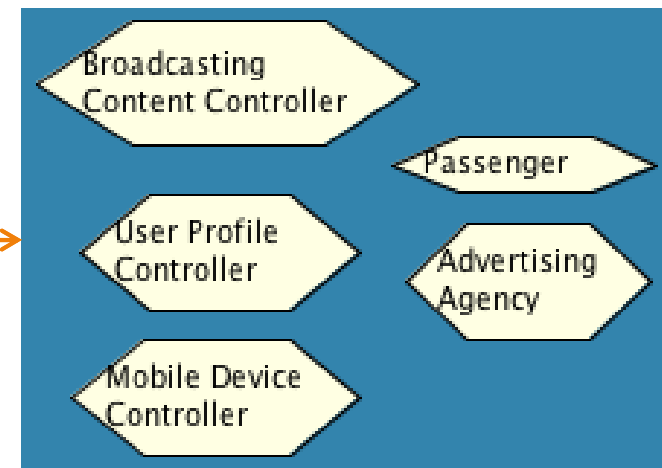
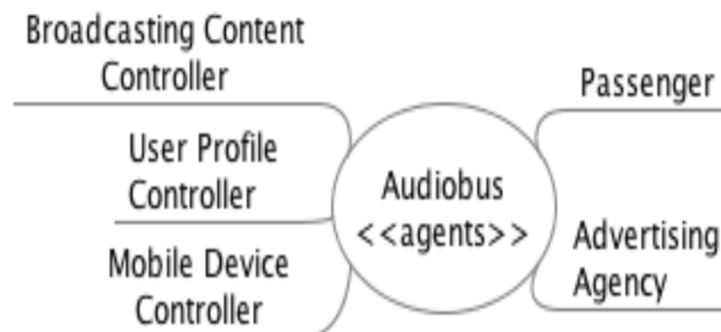
Case Study (*Audiobus*)

54

3rd International Model-Driven Requirements Engineering



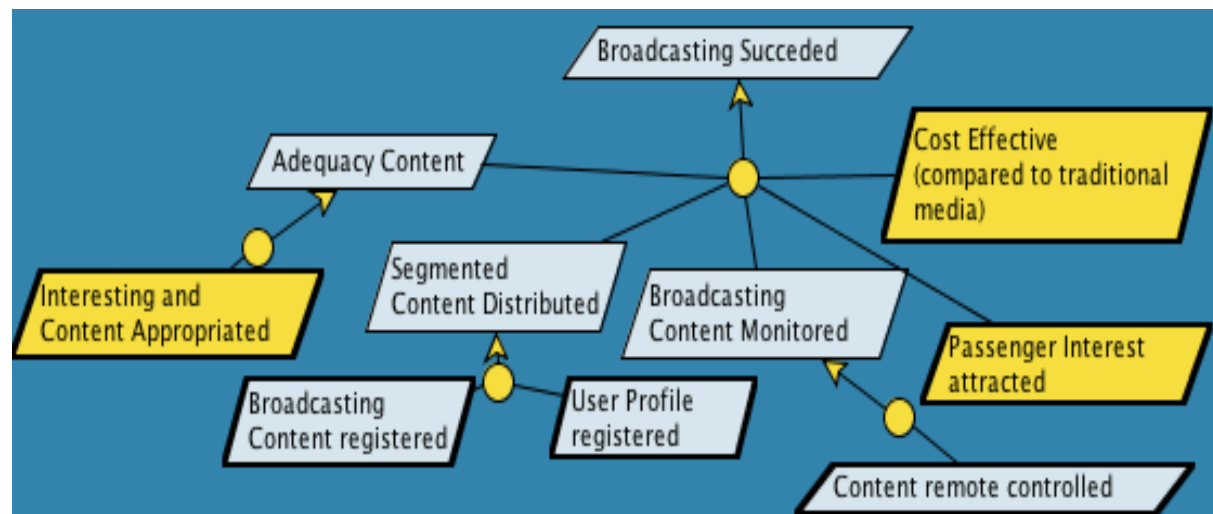
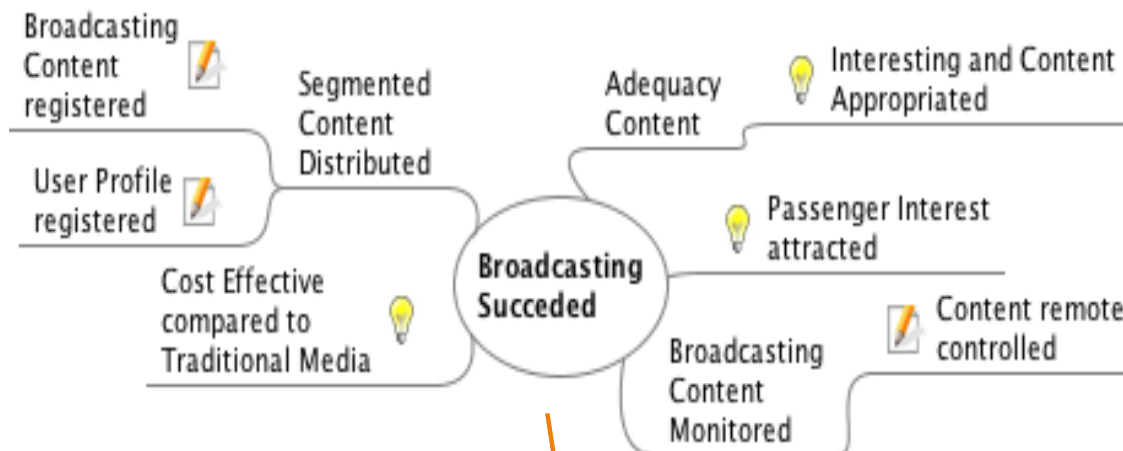
Agents Identification



Goals Identification

56

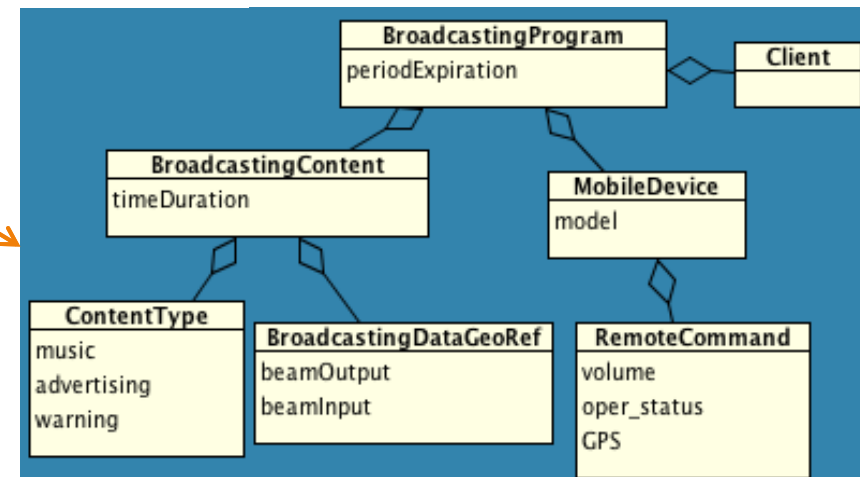
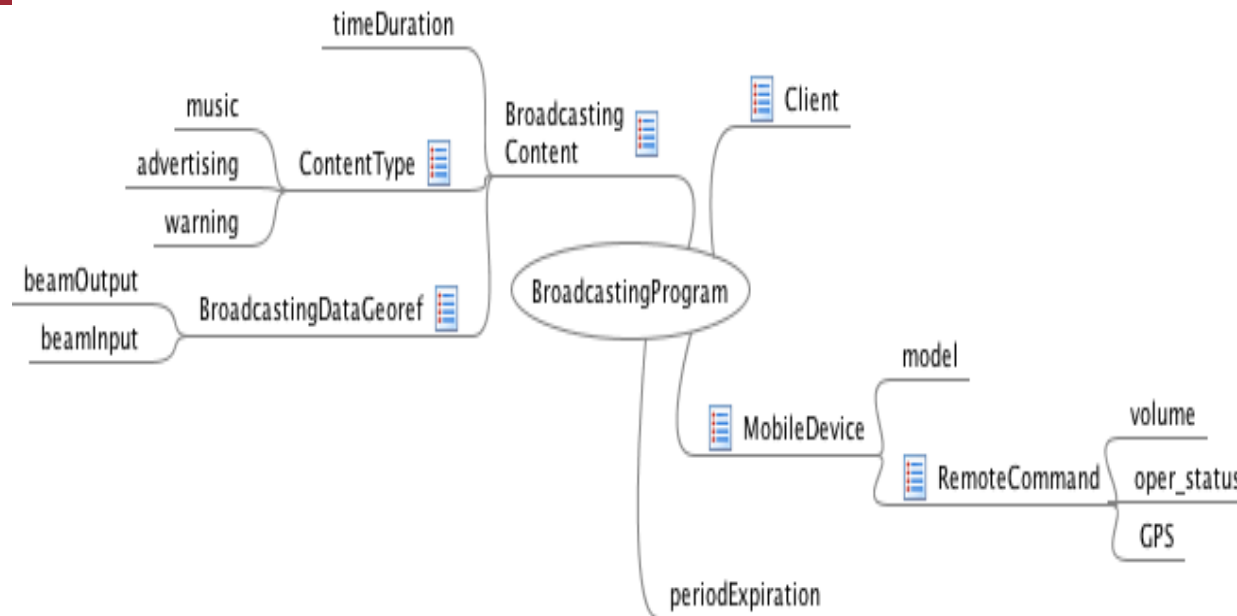
3rd International Model-Driven Requirements Engineering



Object Identification

57

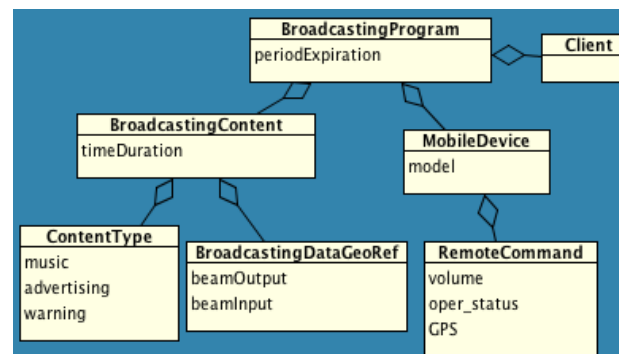
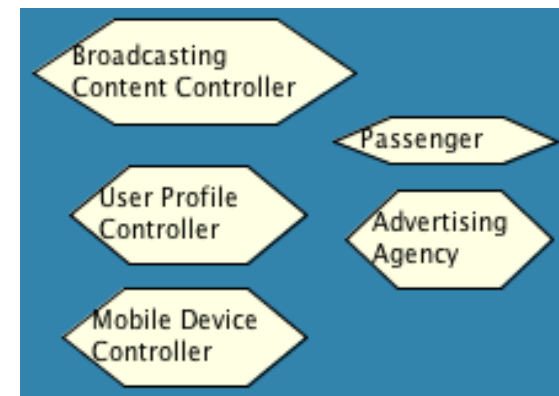
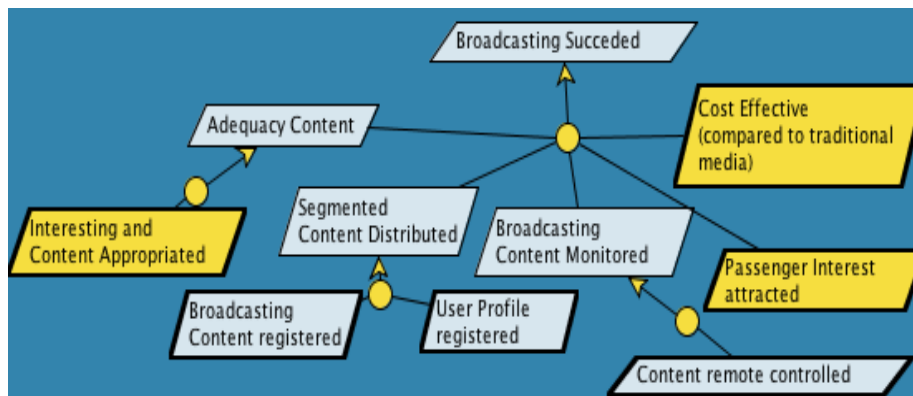
3rd International Model-Driven Requirements Engineering



Compose KAOS Model

58

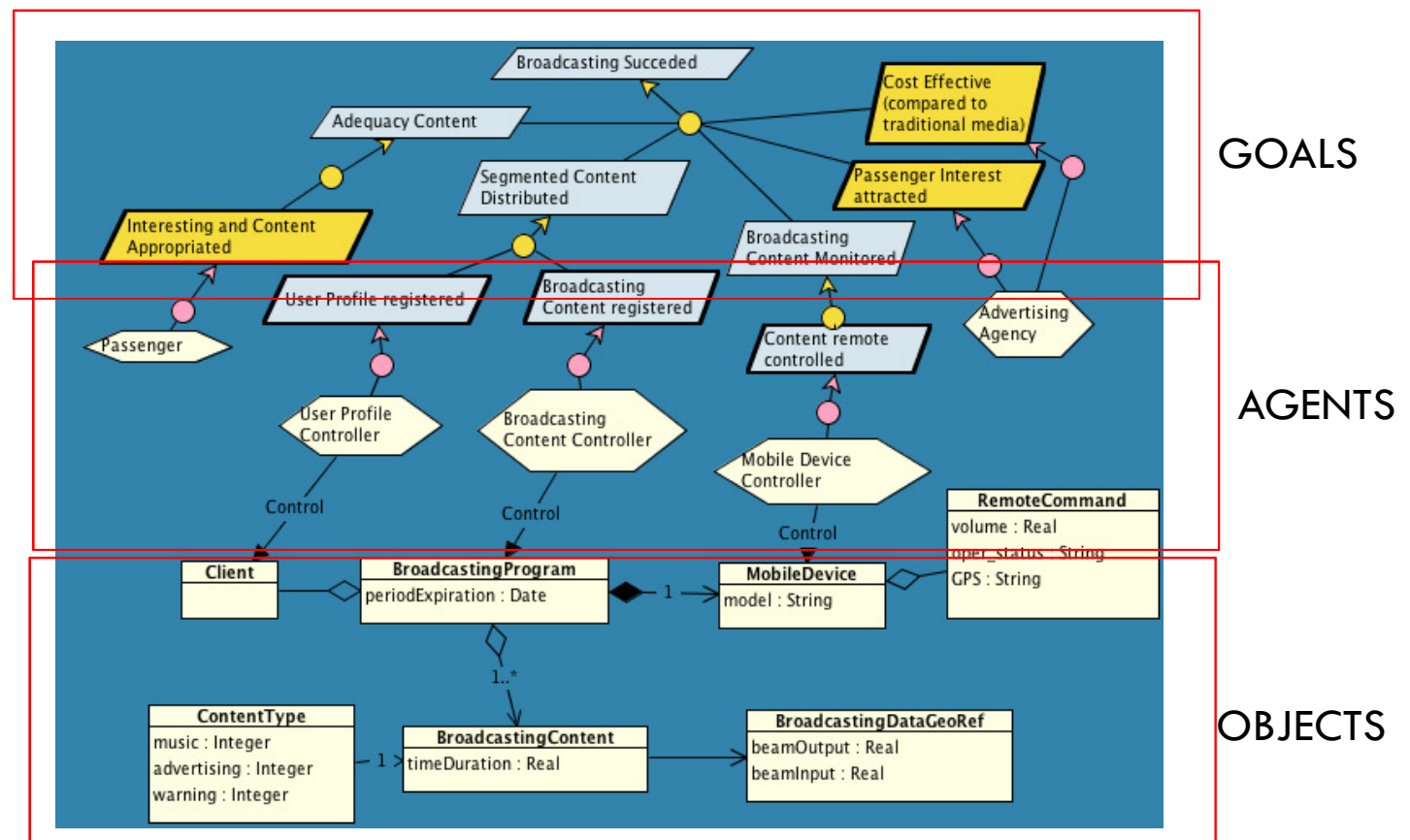
3rd International Model-Driven Requirements Engineering



Compose KAOS Model

59

3rd International Model-Driven Requirements Engineering

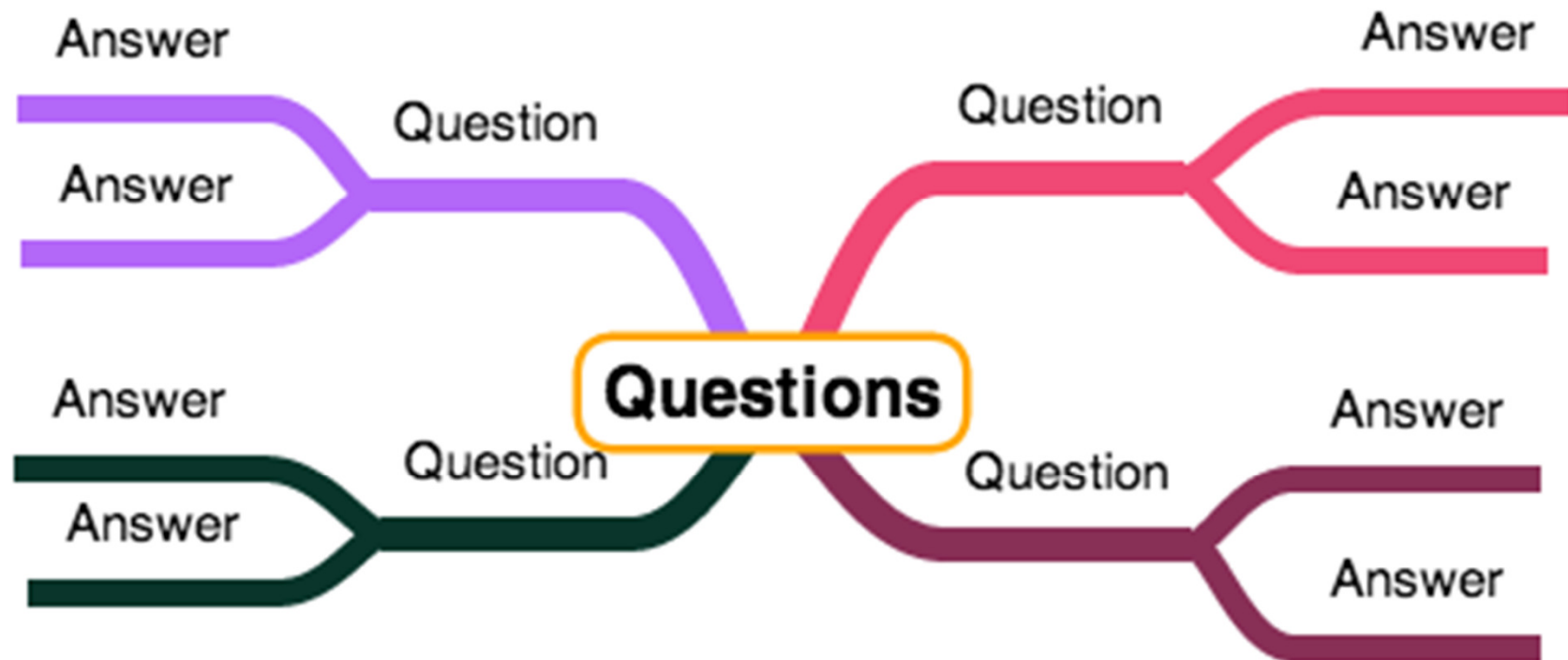


Conclusion

- This work describes a model-driven requirements approach to generate systematically KAOS models from mind maps using meta-modeling and model transformations;
- After the mapping elements (from mind maps) the software engineer composes a complete KAOS model;
- A systematic and agile process was defined for the approach, which was applied to a real case study.

Future Work

- For future work we intend to implement the transformation process with ATL;
- To design an empirical protocol to evaluate the understandability of mind maps adoption and;
- To include mind maps for capturing operations to be included in KAOS models and to automate the composition part.



“Weyuker properties” assessment

#	Adapted Weyuker Property	ANLG	GNO	MD	RNSG
1	At least some different models should exhibit different values for the same complexity metric.	Yes	Yes	Yes	Yes
	$\exists P, \exists Q : P \neq Q \wedge P \neq Q $				
2	There is a finite number n of models for which the complexity is c (a non-negative number). Let S be the set of models with c complexity, and n the cardinal of the set S .	Yes	Yes	Yes	Yes
	$\forall c \in \mathbb{R}_0^+, \forall P : P = c \Rightarrow P \in S, \exists n \in \mathbb{N}_0 : \#S = n$				
3	Different models P and Q may have the same complexity.	Yes	Yes	Yes	Yes
	$\exists P, \exists Q : P \neq Q \wedge P = Q $				
4	Different models which are functionally equivalent may have different complexities.	Yes	Yes	Yes	Yes
	$\exists P, \exists Q : P \equiv Q \wedge P \neq Q $				
5	Monotonicity is a fundamental property of all complexity measures. A model in isolation is at most as complex as its composition with another model.	Yes	Yes	Yes	Yes
	$\forall P, \forall Q : P \leq P; Q \wedge Q \leq P; Q $				

“Weyuker properties” assessment

#	Adapted Weyuker Property	ANLG	GNO	MD	RNSG
6	The resulting complexities of composing the same model (R) with two different models of the same complexity (P and Q) are not necessarily equal. Conversely, the complexities of composing two different models (P and Q) of the same complexity with a third program (R) are also not necessarily equal.	Yes	Yes	Yes	Yes
	$\exists P, \exists Q, \exists R : P \neq Q \wedge P = Q \wedge P; R \neq Q; R $ $\exists P, \exists Q, \exists R : P \neq Q \wedge P = Q \wedge R; P \neq R; Q $				
7	Weyuker’s property 7 states that program’s complexity should be responsive to the order of its statements, and hence to their potential interaction. In a KAOS goal model, the adapted rule would be that the model complexity should be responsive to the organization of its model elements in the goal model graph. Let P be a model and Q another model such that Q is formed by permuting the order of the elements in P. Assume we name this permutation operation Perm().	Yes	Yes	Yes	No
	$\exists P, \exists Q : Q = \text{Perm}(P) \wedge P \neq Q $				
8	If a model is a renaming of another model, then their complexity should be the same. Assume that the operation Rename() transforms model P in its renamed version Q.	Yes	Yes	Yes	Yes
	$\forall P, \forall Q : Q = \text{Rename}(P) \Rightarrow P = Q $				
9	The complexity of the composition of two models P and Q may be greater than the sum of the complexities of models P and Q. The extra complexity may result from the interaction between models P and Q.	No	No	No	No
	$\exists P, \exists Q : P + Q < P; Q $				