Measuring Complexity and Completeness of KAOS Goal Models

Patrícia Espada, Miguel Goulão, João Araújo
CITI, Departamento de Informática
Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa
Lisbon, Portugal
titiespada@gmail.com,
{miguel.goulao, ja}@di.fct.unl.pt

***Abstract*—KAOS is one of the most well-known goal-oriented requirements engineering approaches. Nevertheless, building large KAOS models sometimes results in incomplete and/or complex requirements models that are difficult to understand and maintain. These shortcomings often lead to an increase in costs of product development and evolution. Therefore, for large-scale systems, the ability to manage the complexity and completeness of KAOS models is essential. In this paper, we propose a metrics suite for supporting the quantitative assessment of KAOS models complexity and completeness, in order to support their early identification. We apply the metrics to an example taken from a health club system specification.**

Keywords: Goal-Oriented Requirements Engineering, Requirements Metrics, Model Complexity, Model Completeness.

I. INTRODUCTION

Goal-Oriented Requirements Engineering (GORE) [1] is a paradigm for requirements elicitation where requirements, normally associated with system's stakeholders, are associated to goals. These requirements are identified, analysed and assigned to system components or environmental agents. The main motivations of this vision of Requirements Engineering (RE) are: **(i) communication and understanding** – due to various levels of abstraction obtained by the refinement of goals, GORE provides a comprehensive framework for documenting the requirements, and also an effective way to communicate with stakeholders; **(ii) variability** – through the refinement of alternative goals and alternative assignment of responsibilities, it is possible to explore several configurations of the system; **(iii) completion** – the formalization of the goals allows to prove its completeness and accuracy; **(iv) traceability** – the goals refinement offers vertical traceability from the highest level goals to detailed requirements; **(v) conflict management** – goals can be used to detect and manage conflicts among requirements.

The most well-known GORE approaches are *i** [2] and KAOS [1]. In this work, we focus on the analysis of the complexity and completeness of KAOS models. We target our approach to KAOS models that do not use formal specifications. KAOS is a methodology that aims to support the entire requirements development process. The main steps to building KAOS specifications from high level goals are [3, 4]: **(i) goals development** – goals refinement through the identification of new and more specific goals that characterize the high-level ones; **(ii) objects identification** – identification of objects in the formulation of the goal, definition of the links among them, and description of the domain properties; **(iii) operations identification** – identification of object state transitions that are significant to the goal; **(iv) goals operationalization** – specification of operations in order to satisfying all goals; **(v) responsibilities assignment** – mapping of agents to leaf goals and operations assignment to agents.

Our approach provides a metrics-based analysis framework for KAOS models. Metrics can be valuable to analyse properties such as the complexity and completeness of KAOS goal models. We use the Goal-Question-Metric (GQM) approach to define our metrics set.

GQM [5, 6] is a goal-oriented approach for measuring software systems based on three levels of abstraction. This approach begins by identifying the measurement goals (conceptual level), taking into account the organization, purpose, quality attributes, views and environment. Each goal is then refined into several questions (operational level), which characterize how a particular goal can be achieved. Metrics are then identified (quantitative level). They provide quantitative information to answer questions from the previous level. Fig. 1 shows the structure of the GQM.

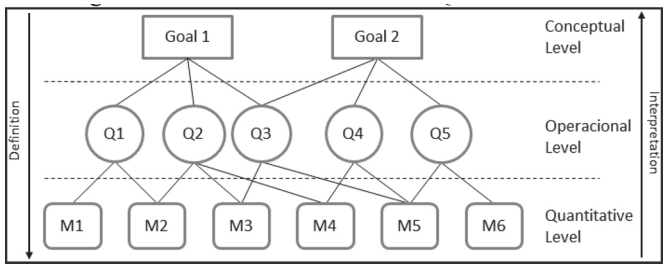

FIGURE 1. GQM STRUCTURE [7]

II. OBJECTIVES OF THE RESEARCH

Our aim is to analyse KAOS models with respect to their completeness and complexity. These are our GQM conceptual-level goals.

Sometimes, KAOS models get so large that is difficult to verify how far the analysts are from completing the model. However, it is also important to manage the model's complexity. It is upon these basic ideas that we propose a set of completeness and complexity metrics over KAOS models. Table 1 summarises the outcome of applying the GQM approach to propose a set of metrics that will allow satisfying

978-1-4577-1076-6/11/$26.00 ©2011 IEEE

the goals of completeness and complexity evaluation. The first column presents the goal to be fulfilled. The second column presents questions that, once answered, will allow satisfying the goals in column 1. The third column presents metrics proposed to answer the above mentioned questions.

TABLE 1 – METRICS DEFINITIONS FOR KAOS GOAL MODELS

Goal	Question	Metric
Completeness	How close are we to completing the assignment of all goal responsibilities to agents?	**M1**. Percentage of leaf goals that have an associated agent.
	How detailed is the goal model with respect to objects?	**M2**. Percentage of leaf goals that have an associated object.
Complexity	Does an agent have too much responsibility in the model?	**M3**. Minimum number of leaf goals associated with an agent.
		M4. Maximum number of leaf goals associated with an agent.
		M5. Average number of leaf goals associated with an agent.
	Does a leaf goal have too many/few associated objects?	**M6**. Minimum number of objects associated with a leaf goal.
		M7. Maximum number of objects associated with a leaf goal.
		M8. Average number of objects associated with a leaf goal.
	How difficult is it to understand a leaf goal, with respect to its parent goals?	**M9**. Depth of the goal hierarchy.
	How complex is a goal, with respect to its refinements?	**M10**. Number of direct or indirect sub-goals of a goal.

Metric M1 directly relates to KAOS models completeness rules [4]. In a complete KAOS goal model, all leaf goals must be assigned to an agent. Metric M2 is concerned with the level of detail provided by the identification of system objects. While they are not mandatory for achieving a complete model, they do provide valuable details to be used in subsequent development stages of the system.

Metrics M3, M4 and M5 helps identifying "good" agents – an agent with too many responsibilities in the system, which is usually a sign of poor modelling. Likewise, we have the metrics M6, M7 and M8 that help realize when leaf goals are too many (or too few) objects. Metric M9 reflects the complexity of understanding a leaf goal. The deeper the goal hierarchy, the harder it is to understand the rationale for the leaf goal. This may lead to an increased effort in the event of requirements change. Metric M10 helps identifying structural problems in goal decomposition. A goal with too many sub-goals should be scrutinized for a potential lack of cohesion.

III. RELATED WORK

Metrics-based assessment of requirements models remains a fairly unexplored topic, although improving the quality of requirements models is expected to have a significant impact in the software development process.

Ramos *et al.* proposed the AIRDoc approach [8]. AIRDoc is a model that aims to facilitate the process of identifying potential problems in requirements documents using refactoring and patterns. AIRDoc also uses the GQM approach for the evaluation of requirements models, namely use cases and respective scenarios descriptions. The target quality attributes were reusability and maintainability, different from ours.

Giachetti *et al.* proposed an approach to evaluate the suitability of *i** models to generate a class model, as a basis for Model-Driven Development (MDD) processes [9]. They defined a set of metrics for *i** models to validate the accuracy of these models. This model also adopts the GQM approach. Compared to ours, their approach focuses on a different set of metrics as their objective was to support the evaluation of *i** models to generate class models.

Franch and Grau in [10] proposed a framework for defining metrics in *i** models, in order to analyse the quality of individual models, as well as to compare alternative models over certain properties. This framework uses a catalogue of patterns for defining metrics, and Object Constraint Language (OCL) to formulate these same metrics. Later, Franch proposed a generic method to better guide the analyst throw out the metrics definition process, over *i** models [11]. This was applied to evaluate business process performance.

While building the complexity metrics in our set, we were also inspired by software complexity metrics proposed for other contexts (e.g. Object Oriented Design [12]).

IV. EXAMPLE

In this paper we will apply the defined metrics to an example. Our aim is to model functionalities of a health club system, such as: control people's access to the health club; track all the members' payments and all the wages paid; classes and courses management; people management; vouchers system, for the members reserve a place in the class; website, that provides a chat system and all the information about the health club.

Next, we will present two goal models for the access control functionality: (i) an incomplete goal model and (ii) a refined version of the goal model where agents and objects are added. Then, we will apply the set of metrics, as listed in the previous section, and draw some conclusions about the results obtained by computing these metrics upon these goal models.

A. Model 1: Access Control Model

The access control is an essential functionality of the health club example. As expected, the model presents the "Register entry", "Register exit" and the "Card verified" goals. Associated to the first two goals we have the objects entries, exits and members. Unlike these goals, the "Card

978-1-4577-1076-6/11/$26.00 ©2011 IEEE

verified" goal does not have objects associated to it. Also, agents are missing to guarantee the model's completeness.

The model consists of 18 direct or indirect refinements from the original goal and, among these, 9 are leaf goals.

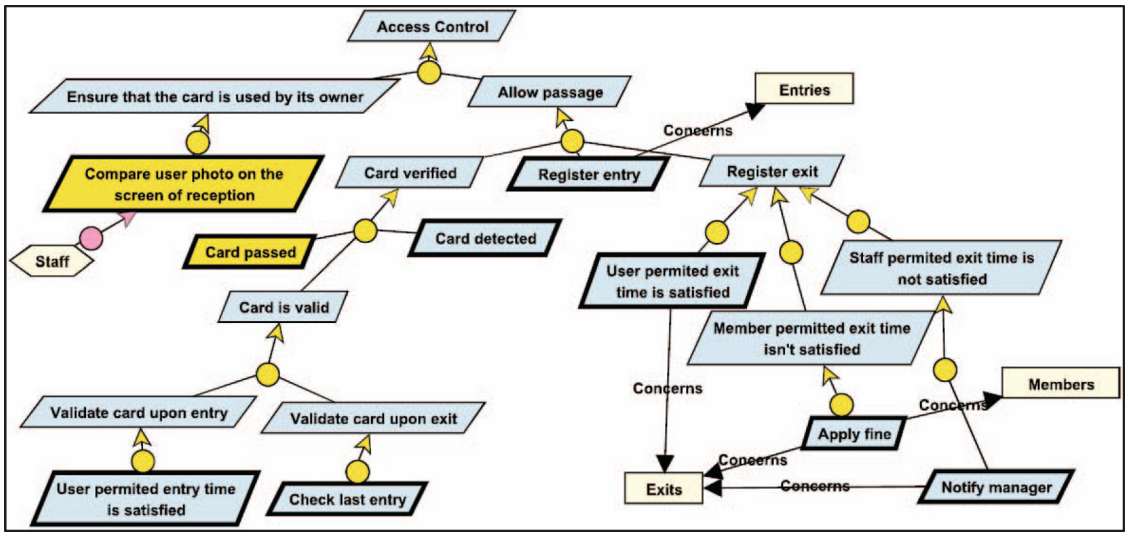

FIGURE 2. ACCESS CONTROL GOAL MODEL

B. *Model 2: Refined Access Control Model*

Our aim in this subsection is to show a more complete model than the one in Fig. 2. Thus, taking into account the above observations, we enriched the model by introducing the following elements, as can be seen in Fig. 3: **(i) environment agents** – user and card reader; **(ii) system agents** – access controller agent to help the card verification process, and a registration controller; **(iii) objects** – cards.

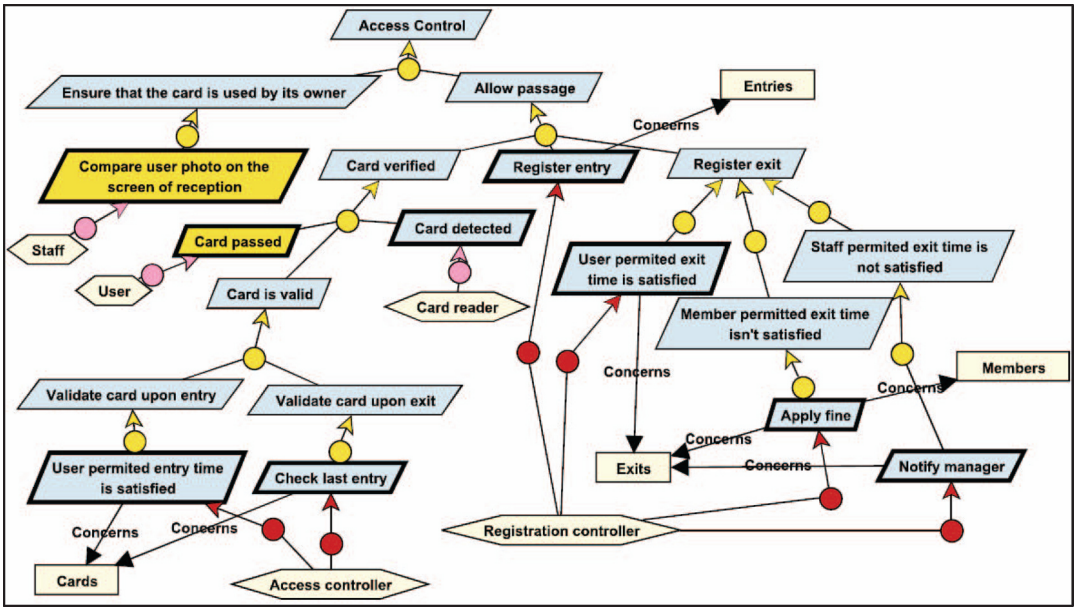

FIGURE 3. REFINED ACCESS CONTROL GOAL MODEL

C. *Metrics Analysis and Discussion*

Next we show how the addition of agents and objects to the first model can improve the model's completeness, and how it impacts on the model's complexity. Table 2 shows the result values of applying the set of metrics defined in Section II to both models.

Upon the values presented in Table 2, we can draw the following observations:

- By adding a significant number of agents, we have a positive increase of completeness from the first model to the second one (as shown by the metric M1).
- In the values of the metric M2 we can see an increase of about 23% in the completeness between the two models. More objects indicate a more detailed model.
- The first model has only one agent, which results in a constant value of metrics M3, M4 and M5. However, by

978-1-4577-1076-6/11/$26.00 ©2011 IEEE

adding more agents to the second model, we can see an increase of responsibility of each agent.

- Although metrics M6 and M7 are equal in both models, we have an increase of the average number of objects associated with each leaf goal (M8). This means that there are more leaf goals with at least one associated object, making the model more complete, but also more complex.
- As expected, metrics M9 and M10 show no difference between the two models. But if the values are considered high, model refactoring can be applied.

TABLE 2 – METRIC ANALYSIS OF THE GOAL MODELS

Metric	Value	
	Model 1	**Model 2**
M1. Percentage of leaf goals that have an associated agent.	≈ 11%	100%
M2. Percentage of leaf goals that have an associated object.	≈ 44%	≈ 66,7%
M3. Minimum number of leaf goals associated with an agent.	1	1
M4. Maximum number of leaf goals associated with an agent.	1	4
M5. Average number of leaf goals associated with an agent.	1	1,8
M6. Minimum number of objects associated with a leaf goal.	0	0
M7. Maximum number of objects associated with a leaf goal.	2	2
M8. Average number of objects associated with a leaf goal.	≈ 0,56	≈ 0,78
M9. Depth of the goal hierarchy.	5	5
M10. Number of direct or indirect refinements of a goal.	18	18

In short, by adding significant objects and agents to the Access Control model we affect the completeness in a very positive way. On the other hand, the impact of these changes on the complexity cannot be considered harmful, as the average number of agents' responsibilities is acceptable (and also taking into account the minimum and maximum value (metrics M3 and M4)).

There are three main concerns that our metrics want to prevent and inform. The first is to guarantee that the completeness must be achieved by increasing the number of agents and objects. The second is to guarantee that the complexity does not suffer a negative impact by the addition of more objects and agents. The third is to ensure that the number of responsibilities for each agent is reasonable, avoiding situations where we have lazy agents or agents with too many responsibilities.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed the definition of a metrics set for assessing the completeness and complexity of KAOS goal models. Completeness analysis is useful to help requirements engineers to check the extent to which their models are close to being complete. Completeness is an essential property, as the lack of it at early stages may ultimately lead to extra costs in the development process. Complexity analysis is particularly useful for identifying issues with the quality of the produced models. In particular, it can be used to help identifying opportunities for requirements refactoring.

We plan to apply these metrics to larger models, in order to validate their usefulness. We also plan to extend the set of metrics to cover other model quality attributes. Finally, we will provide tool support for these metrics.

ACKNOWLEDGMENTS

The authors would like to acknowledge the AMPLE project and CITI – PEst -OE/EEI/UI0527/2011, Centro de Informática e Tecnologias da Informação (CITI/FCT/UNL) - 2011-2012) – for the financial support for this work.

REFERENCES

[1] Lamsweerde, A. v. "Goal-Oriented Requirements Engineering: A Guided Tour". 5th IEEE International Symposium on Requirements Engineering, Toronto, Canada, IEEE Computer Society, 2001.

[2] Yu, E. "Modelling strategic relationships for process reengineering". Thesis (Ph.D.), University of Toronto, Toronto, Canada, 1995.

[3] Lapouchnian, A. "Goal-Oriented Requirements Engineering: an Overview of the Current Research". University of Toronto, Toronto, Canada, 2005.

[4] Lamsweerde, A. v. and Letier, E. "Handling Obstacles in Goal-Oriented Requirements Engineering". *IEEE Transactions on Software Engineering*, 26, 10 (October 2000), 978-1005.

[5] Solingen, R. v. and Berghout, E. *The Goal/Question/Metric Method: a Practical Guide for Quality Improvement of Software Development*. McGraw Hill, England, 1999.

[6] Esteves, J., Pastor, J. and Casanovas, J. "Measuring Sustained Management Support in ERP Implementation Projects: a GQM Approach". *8th Americas Conference on Information Systems (AMCIS)*, Dallas, USA, 2002, pp. 1381-1389.

[7] Basili, V. R., Caldiera, G. and Rombach, H. D. "The Goal Question Metric Approach". *Encyclopedia of Software Engineering*, J. J. Marciniak: John Wiley & Sons, Inc., 1994, pp. 528-532.

[8] Ramos, R., Castro, J., Araújo, J., Moreira, A., Alencar, F., Santos, E. and Penteado, R. "AIRDoc–An Approach to Improve Requirements Documents". *22th Brazilian Symposium on Software Engineering (SBES'08)*, Brazil, SBES, 2008.

[9] Giachetti, G., Alencar, F., Xavier, F. and Pastor, O. "Applying i* Metrics for the Integration of Goal-Oriented Modeling into MDD Processes". Universitat Politècnica de Catalunya, Barcelona, Spain, 2010.

[10] Franch, X. and Grau, G. "Towards a Catalogue of Patterns for Defining Metrics over i* Models". 20th International Conference on Advanced Information Systems Engineering (CAiSE '08), Springer-Verlag, 2008.

[11] Franch, X. "A Method for the Definition of Metrics over i* Models". 21st International Conference on Advanced Information Systems Engineering, Amsterdam, Netherlands, Springer-Verlag, 2009.

[12] Chidamber, S. R. and Kemerer, C. F. "A Metrics Suite for Object Oriented Design". *IEEE Transactions on Software Engineering*, 20, 6 (June 1994), 476-493.

978-1-4577-1076-6/11/$26.00 ©2011 IEEE