USING VARIANTS IN KAOS GOAL MODELLING

Joël Brunet, Farida Semmak, Régine Laleau
LACL (Laboratoire d'algorithmique, complexité et logique), Université Paris XII
61 av Général de Gaulle 94000 Créteil, France

Christophe Gnaho
Université Paris V, 45 rue des Saints-Pères, 75006 Paris, France

Keywords: Goal modelling, variability, requirements engineering.

Abstract: In this paper we apply a certain kind of variability in KAOS goal and responsibility models: links can be conditioned depending on the choice of variants. These variants are grouped in facets and organized in a variant tree having a metagraph semantics: instances are all the goal/responsibility graphs generated when all variants are fixed. We use a case study from the land transportation domain: a simplified cycab (or cybercar) with some variants. The overall case study is part of the ANR TACOS project, whose final aim is to define a component-based approach to specify systems with high-level safety requirements.

1 CONTEXT

Poor requirements have been recognized to be a major cause of software problems such as cost overrun, delayed delivery or failure to meet expectations (The Standish Group, 1995). The problem gets even more serious in the case of safety-critical or security systems; most severe failures have been recognized to be traceable back to defective specification of requirements (Lutz, 1993), (Safety-Critical Systems, 2002). In order to take into account these important limitations, Goal-Oriented Requirements Engineering (GORE) is concerned with the elicitation of the goals to be achieved by the system envisioned – WHY – issues, the operationalization of goals into specifications of services and constraints – WHAT issues –, and the assignment of responsibilities to agents such as humans, devices and software pieces available or to be developed – WHO issues – (van Lamsweerde, 2003). Paradigms using the goal concept have been proposed by several approaches: KAOS (Dardenne, 1993), I* (Yu, 1997), CREWS (Rolland, 1998). We choose the KAOS approach for different reasons including the presence of the Objectiver CASE tool and a certain effectiveness of the approach.

The work presented in this paper is part of the TACOS project (**T**rustworthy **A**ssembling of **C**omponents: fr**O**m requirements to **S**pecification, funded by the French Research Agency – *Agence Nationale de la Recherche* – under the ANR-06-SETI-017 reference), which started in January 2007, and whose aim is to define an engineering process beginning with functional and non-functional goals and ending with formal specifications organized as components that check some properties like security, efficiency, fault tolerance, interoperability, etc. Land transportation has been chosen as the application domain of the project. More specifically we focus on the new transportation systems named cybercars or cycabs, that were and are always the subject of several research projects, such as CyberCars and Cybermove (Cybercar projects).

In this paper, we study how a certain kind of variability – called by us variant trees – can be used in KAOS goal models. We illustrate our approach on a simplified cycab case study that could be summed up as follows (many assumptions are still left open, some of them will be treated by introducing some variants).

Simplified Cycab Case Study. *We consider a unique cycab, filoguided on a dedicated road and servicing a succession of stations where passengers can get in and get off. After the last station, the cycab goes to the first one (because it is on a circular road). The cycab cannot turn round.*

From our point of view, the purpose of variability is to consider and represent the large diversity of options a given application of a given domain may take. Variability has already been studied in the domain of software product lines (van Gurp, 2001) (Halmans, 2003). It can be defined as the ability of an element (component, system, model…) to be changed, personalized and configured according to a specific context. Bachmann and Bass (2001) propose to classify variability into several categories (functions, data, technology...). Halmans and Pohl distinguish between essential variability for functional and non-functional needs and technical variability for implementation. Variability is represented in (Jacobson, 1997) and (Halmans, 2003) by variation points and variants on use cases: a variation point defines a point in the model where variation occurs, whereas a variant is a manner of realizing variability. Moreover, mechanisms such as optionality, alternative and optional alternative to organise them are used (Bachmann, 2001), (Halmans, 2003).

Variability has also been studied in domain analysis. Among domain analysis methods (Arango, 1994), the FODA method (Kang, 1998) has been the first one to propose the concept of *feature*, defined as *a prominent or distinctive user-visible aspect, quality or characteristic of a software system or systems*. The feature model highlights, in the form of hierarchy sets, the characteristics that discriminate systems in a domain. Other approaches study variability at an early requirement engineering step. In Crews project (Bennasri, 2004), task and strategies alternatives are defined in labelled directed graphs called maps (Rolland, 1998), that provide support in alternative selection through guidelines. In (Liaskos, 2006), variability is tackled through OR decomposition of goals, by stating variability concerns that use semantic frames based on linguistic invariants, and with a mechanism of background variability. This mechanism allows to describe characteristics of agents, locations and objects of the domain, that may lead to identify additional alternatives. A survey of variability in requirement engineering can be found in (Liaskos, 2007).

The originality of our approach is to apply variability to goals, not only to analysis, design or programming products. Our approach of variability is in continuation of our previous work (Semmak, 2005), (Semmak, 2006) except that we now use a well-known goal approach (KAOS) rather than a specific one.

In section 2 we introduce the notion of variant that seems necessary in respect with our objectives. We apply it on goal and responsibility models and sketch out the impacts on the other KAOS models. In section 3 we group together variants in facets, organize them in a tree called variant model or graph, and present a metamodel to relate variant models with goal models. Finally we conclude in section 4.

2 VARIANTS

2.1 Variants applied to Goal Models

In KAOS, a goal can be reduced by AND decomposition; that means that all the subgoals are necessary to satisfy a goal. Moreover, a goal may be reduced by OR decomposition that allows several alternatives to satisfy a goal to be stated. It is important to observe that OR decompositions are placed before AND decompositions: we can say that a goal is satisfied either by this set of subgoals or by this other one. If we want the contrary, we must introduce intermediary goals: a goal is achieved by the satisfaction of all its subgoals, each of them being possibly decomposed with alternatives. Another remark is that alternatives may be needed not only for low-level goals but also for abstract soft goals. For instance, if we study all kinds of public libraries, we can find some public libraries that allow documents to be consulted and borrowed, while others allow only documents to be consulted or only documents to be borrowed. Thus, a root goal "documents put at disposal" may be reduced by an OR decomposition with the subgoals "documents put in consultation" and "documents put in borrowing". Each of these subgoals will share some common subgoals such as "documents managed", that is the reason why goal models are not trees even if there is a root: they just are oriented graphs with a root.

In order to justify the introduction of the variant concept, we may highlight that some variations in the requirements cannot be simply represented by alternatives, because it may have an impact on different parts of the goal graph. By using an alternative, it seems that what we can only do is to determine the smallest subgraph that contains all the impacts of the variation, to duplicate this subgraph, to introduce the variations on one of them, and to link both of them with an OR link to their parent goal. When several variations are introduced in the requirements, this mechanism leads quickly to combinatory explosion in goal models, not in terms of goals but in terms of links between goals. One can easily imagine that only a few variations can be represented if we want to keep the goal model readable.

Thus, we argue that the notion of variant, that we define as *some requirement that may or may not be included in the final set of requirements we want to be satisfied by our future system*, can produce a more simple goal model by "factorizing" identical links between several variants.

For instance, let us consider the following variants concerning the moving mode of the cycab: it can either be automatic (for instance a subway stopping at every station) or on demand (it stops at a station only if there is an external or internal demand, such as an elevator). In order to annotate the goal model while preserving clarity, we associate boolean variables to these variants: respectively AUTOMOV (automatic moving) and ONDEMMOV (on demand moving). Figure 1 presents the main part of the goal model of the case study. The semantics of a variable affected to a link is: *the link is valid if and only if the value of the variable is true.*

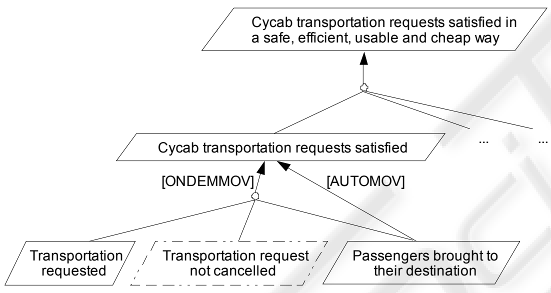
Cycab transportation requests satisfied in a safe, efficient, usable and cheap way
Cycab transportation requests satisfied
...
...
[ONDEMMOV]
[AUTOMOV]
Transportation requested
Transportation request not cancelled
Passengers brought to their destination

Figure 1: Root goal model of the cycab case study.

Commentaries: the dashed box is for an hypothesis; we do not use a specific notation to distinguish between goals and requirements because the latter may be identified by their attached responsible agent (when all of them are affected); the suspension points are for non-functional goals.

In natural language, the goal "Transportation requested" and its corresponding hypothesis are only relevant for the variant *On demand moving*. One can obviously object that, in figure 1, our variant notion is not necessary: it may be tackled by an alternative. We agree, but it gives a semantics to the alternative because, as we will see later, variants may be organized and documented. Moreover there is a more important reason: if a variant has a consequence in another part of the graph, what we just have to do is to use the same variable to condition a new link. For instance in figure 2, when the goal "Passengers brought to destination" is reduced, we find a new impact of these variants: the subgoal "Destination selected" has a meaning only if the variant *On demand moving* is selected.

In this example we can see that if an alternative decomposition is used, the number of links in the graph will almost double (if we enumerate distinctly the aggregate part of the links and their membership parts, that are respectively above and under the blank circle). It seems useful to place variants either in the aggregate part of an AND link or in the membership part of it, despite the fact that they could be put only in the membership parts. We prefer not to be restricted to that for the reason of model readability: for instance on figure 1 the second member of the alternative should be deleted and the two first subgoals should be conditioned with the variant [ONDEMMOV], while the variant [AUTOMOV] disappears.

2.2 Variants applied to Responsability Models

In order to experiment the approach on the responsibility model, we have been interested in other variants: the driving mode, either automatic (control driving system) or manual (human driver), and the system chosen for opening and closing the doors of a cycab, that could be either automatic or manual (in a simplified approach). We associate to these four variants the respective boolean variables AUTODRIVE, MANDRIVE, AUTODOOR, MANDOOR.

The impact of these variants is shown in the two following figures. The semantics of the annotation of a responsibility link with a variant is similar to the one on a goal reduction link in the previous section: if the variant is true, the responsibility link is either

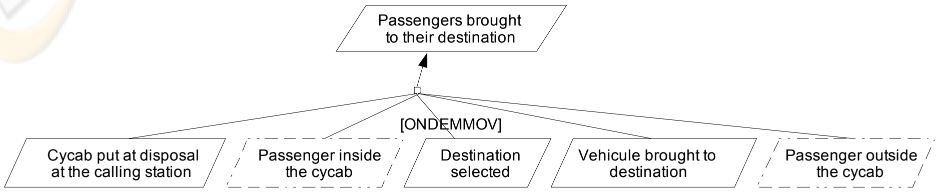
Passengers brought to their destination
[ONDEMMOV]
Cycab put at disposal at the calling station
Passenger inside the cycab
Destination selected
Vehicule brought to destination
Passenger outside the cycab

Figure 2: Goal submodel of the goal Passengers brought to destination.

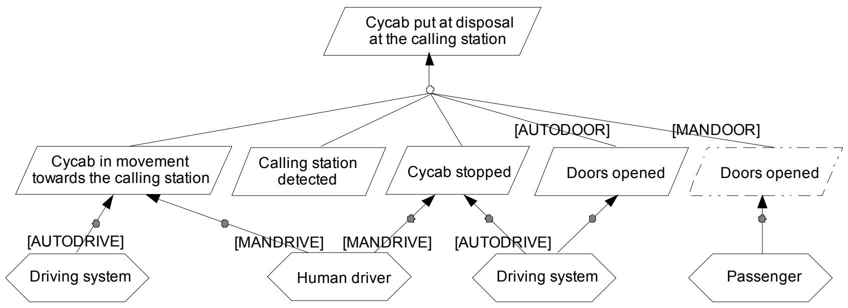

Figure 3: Goal submodel of the goal Cycab put at disposal at the calling station.

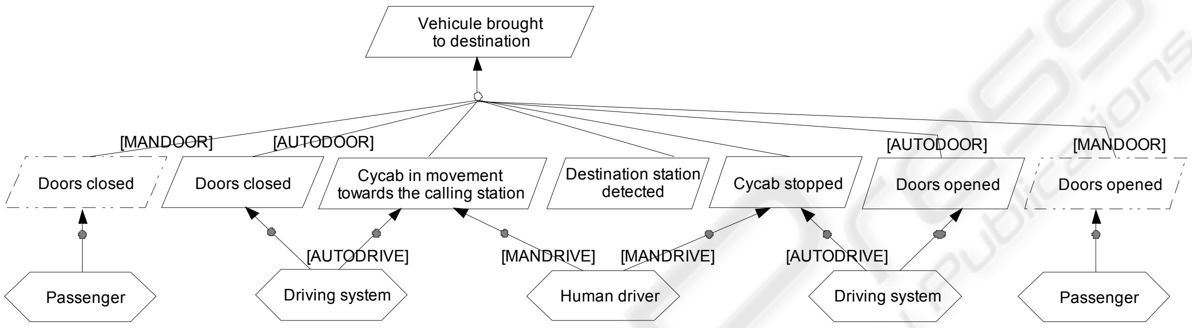

Figure 4: Goal submodel of the goal Vehicule brought to destination.

used, or rejected.

On figure 3 another case is presented: a requirement transformed into an hypothesis depending on the choice of a given variant.

Figure 4 does not introduce new matters but shows that the proposed mechanism can be generalized and how variants impact on the different subgraphs of a goal model. In order to reduce the complexity of the graph, we could have used a graphical formalism to transform conditionally a requirement in an hypothesis in the same box, but this formalism would complicate unnecessarily the graphical conventions.

2.3 Variants applied to other KAOS Models

Applying our variant mechanism to links of other KAOS models (namely operation and object models) has not be considered for the moment. Indeed, it is already possible to infer interesting statements by applying some semantic transitivity rules. When a variant is applied on a reduction link between a goal and a requirement, it may be deduced that the variant puts a condition not only on the requirement but also on the objects and operations linked to it respectively by *Concerns* links and *Operationalizes* links. With the same principle, when a variant is applied on a responsibility link between a goal and an agent, it may be deduced that the variant puts a condition not only on the agent but also on the operations that the agent may possibly execute. For example let us examine the [ONDEMMOV] variant conditioning the reduction link to the requirement Transportation requested in figure 1. If an object Calling button is related through an operationalization link to this requirement, it may be deduced that a calling button is needed if the chosen moving mode is *on demand*.

3 FACETS AND VARIANT GRAPHS

When we thought about variability in the simplified cycab case study, we easily found three variability examples that have a direct impact on the goal model:

- the moving mode that may be automatic or on demand,
- the driving mode, either automatic or manual (with a human driver),
- the opening doors system, either automatic or manual.

We began to organize them in a simple tree, in which certain nodes were named facets (moving mode, driving mode, opening doors system) and others were variants. We defined a facet as *a choice to be made among several variants*. Facets seem to be possibly linked together with an AND semantics, because each facet has to be fixed for a given system, while variants seem to be possibly linked together with an OR semantics, because one of the variants has to be chosen. It's important to notice that exclusive-OR semantics is too strong to represent such a piece of knowledge, on the same manner as for KAOS OR-decomposition links. For instance the moving mode may vary depending on time considerations (rush periods), number of users demands, presence of other cycabs, etc. Thus the variants *automatic moving mode* and *manual moving mode* may be both effective on a system. It is the same for the driving mode, depending for instance on parts of the lap or on different functioning modes (in the evening, cycabs may return alone to their night station in an automatic slow mode). The opening doors system of one cycab cannot be both automatic and manual, but we can consider that heterogeneous cycab could be used on the same lap, ones having automatic doors and others manual doors.

Facets and variants can be organized in a goal graph if they are transformed into goals by adding to them the passive verb "fixed" with the semantics "chosen". This gives the following semantics to these new goals:

- a "variant" goal is satisfied if and only it is fixed for a given system to build, by fixing its boolean value (true="*we keep it*", false="*we don't use it*"),
- a "facet" goal is satisfied if and only if one of its variants is fixed,
- all facets are attached to an higher-level goal named "variants fixed" through an AND link, because a given system is fixed if and only if all its associated facets are fixed.

Thus, the goal model of figure 5 was produced. We called it the variant model (or variant graph) of the goal model of the case study.
The meaning of this tree is partially related to the construction process of the requirements: each facet when fixed by the engineering team will have a precise impact on the corresponding goal model.
Two complementary uses of this kind of graph may be considered:

- some parts of the graph and the consequences on the goal and responsibility models are already specified in a domain repository; impacts of variants may be studied and used as some facets for choosing or rejecting variants,
- other parts of the graph and the consequences on the goal and responsibility models are established by the engineering team, as an help in visualizing and rationally choosing the adapted variants that fit well with the main goal to satisfy.

Obviously there may be more than two variants in a facet: for instance we may add to the *Driving mode fixed facet* the variant *Distant visiophonic driving mode fixed* (used only in restricted cases).
Now we present how to modify the partial metamodel of KAOS (see e.g. Heaven 2004). Two preliminary modifications are needed. Firstly, we transform the responsibility and reduction links into associative objects (represented in figure 6 as classical objects) in order to allow them to be destinations for variants.

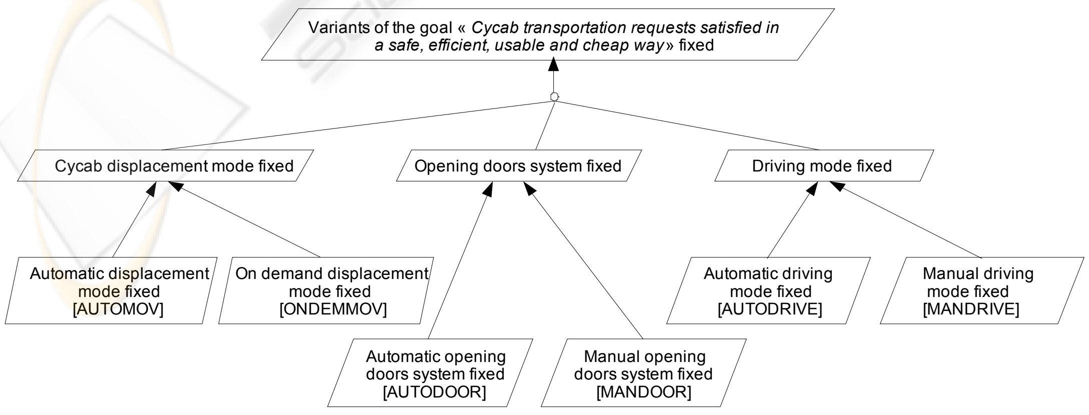
Variants of the goal « Cycab transportation requests satisfied in a safe, efficient, usable and cheap way» fixed
Cycab displacement mode fixed
Opening doors system fixed
Driving mode fixed
Automatic displacement mode fixed [AUTOMOV]
On demand displacement mode fixed [ONDEMMOV]
Automatic opening doors system fixed [AUTODOOR]
Manual opening doors system fixed [MANDOOR]
Automatic driving mode fixed [AUTODRIVE]
Manual driving mode fixed [MANDRIVE]

Figure 5: Variants model of the case study.

Secondly, we transform the newly created reduction object into two distinct objects: *Reduction link* and *Conjunctive cluster*, in order to allow the representation of AND-reduction links as well as OR-reduction links, and to allow both of them to be set as destinations for variants.
Conjunctive clusters state the structure of AND-reduction links: they are associated to all the reduction links that refer to the subgoals of the AND cluster.

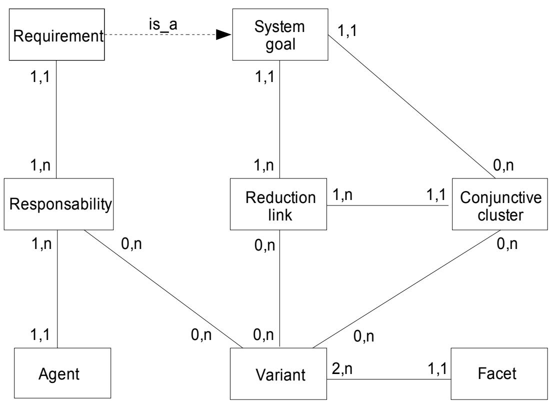

Figure 6: Partial metamodel including variants.

4 FUTURE WORK

Further investigations have still to be done, including: specification of constraints between variants (incompatibilities, etc.), links to operation and object models, and elaboration of guidelines in order to help in the construction of such variants graphs and variant-controlled goal models, and to facilitate the elaboration process of decisions upon the variants choices.
Hereafter are some benefits of variants applied to goals. Firstly, it adds some kind of polymorphism to goal models. Then, it can represent variations that cannot be easily represented with KAOS alternatives. Finally it helps in choosing between options in the expression of a given problem to solve, by allowing consequences on goal and responsibility models to be visualized.

REFERENCES

Arango G., 1994. Domain Analysis Methods, in *Software Reusability*, Eds by W. Schäfer, R.Prieto Diaz & M. Matsumoto, Ellis Horwood.

Bachmann F., Bass L., 2001, *Managing variability in software architecture*, ACM Press.

Bennasri S., Souveyet C., and Rolland C., 2004. Modeling variability in requirements with maps, *Advances in Information Systems*.

Cybercar projects, http://www.cybercars.org/.

Dardenne A., van Lamsweerde A. and Fickas S., 1993. Goal-oriented Requirements Acquisition, *Science of Computer*.

Halmans G., Pohl K., 2003. Communicating the variability of a software product family to customers, *Software and System Modeling*, Springer-Verlag.

Heaven W., Finkelstein A., 2004. An UML profile to support requirements engineering with Kaos, *IEE Proceedings Software*, vol. 151 10-27.

Jacobson I., Griss M., Johnsson P., 1997. *Software Reuse: Architecture, Process and Organization for Business Success*, Addison Wesley.

Kang K., Kim S., Lee J. et al., 1998. FORM: A Feature-Oriented Reuse Method with Domain-Specific Reference Architectures, *Annals of Software Engineering*, 5 143-168.

Liaskos S., Lapouchnian A., Yu Y., Yu E. and Mylopoulos J., 2006. On Goal-based Variability Acquisition and Analysis, *14th IEEE Int. Conf. on Requirements Engineering*.

Liaskos S. & al., 2007. Exploring the Dimensions of Variability: a Requirements Engineering Perspective**,** *Int. Workshop on Variability Modelling of Software-intensive Systems*.

Lutz R., 1993. Analysing Software Requirements Errors in Safety-Critical, Embedded Systems, *First IEEE International Symposium on Requirements Engineering* 126-133.

Rolland C., Souveyet C., BenAchour C., 1998. Guiding Goal Modelling Using Scenarios, *IEEE Transactions on Software Engineering*, Special issue on scenario Management 1055-1071.

Safety-Critical Systems: Challenges and Directions, 2002. ACM Press, *24th International Conference on Software Engineering* 547-550.

Semmak F., Brunet J., 2005. Un métamodèle orienté buts pour spécifier les besoins d'un domaine, *22e INFORSID conference*, 115-132, Hermès Ed., Grenoble, France.

Semmak F., Brunet J., 2006. Variability in Goal-oriented Domain Requirements, *9th Int. Conference on Software Reuse (ICSR)*, LNCS vol. 4039, Springer Verlag.

The Standish Group, Chaos, 1995. Standish group internal report.

van Gurp J., Bosch J., Svahnberg M., 2001. On the notion of variability in Software Product Lines, *Proceedings of the Working IEEE/IFIP Conference on Software Architecture.*

van Lamsweerde A., 2003. From Systems Goals to Software Architecture, *Formal Methods for Software Architectures*, LNCS vol. 2804, Springer.

Yu E., 1997. Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering, *3rd IEEE Int. Symposium on Requirements Engineering* 226-235.