Abstract—In this paper we have modelled an illustrative subset of the requirements of the bCMS case study using a combined approach called SysML/KAOS. This submission is part of a broader effort that aims at the comparison of those three specific requirements modelling techniques (see [1], [2], [3], [4]).

Keywords—goal-oriented requirements; SysML; functional requirement; non-functional requirements

I. INTRODUCTION

The OMG has recently proposed SysML [5], an UML extension for complex systems. This language has some interesting advantages. It provides concepts for linking requirements to the design and the implementation elements, thus ensuring continuity from requirements analysis to implementation. Since it is a UML profile, it offers other benefits like the ability to quickly and easily develop open source tools to support creation of models. However, the SysML concepts for requirements specification are not as rich as those proposed in goal-oriented approaches. We think that it would be interesting to take advantage of the contribution of SysML while providing a clear definition of requirements and relationships between them. The SysML/KAOS [6], [7], [8] language tries to provide an answer to this issue.

The SysML/KAOS language is an extension of the SysML requirements language [5] with the most relevant concepts of KAOS and NFR Framework, two approaches largely recognized and used in requirements engineering over the past decade. The main idea is to combine the concepts of KAOS [10] that are better suited to represent functional requirements with concepts of the NFR model [9] which are most relevant to specify non-functional requirements. This allows the integration of non-functional requirements much earlier at the same level of abstraction than functional requirements.

There are a number of advantages to use Goal Oriented Requirements Engineering (GORE) approaches. They allow detection and resolution of conflicts among requirements and they help reasoning about alternative configurations. In addition to the above advantages, our approach specifically allows the detection and analysis of impact of non-functional requirements on functional requirements.

Fig. 1. bCMS Case Study Overall View

I. INTRODUCTION

The paper is organized as follows. In the following subsections, we present the bCMS case study as well as the minimum of background needed to understand the models. Section II describes the approach and Section III the bCMS requirements model obtained with the SysML/KAOS approach. After the conclusion, the detailed requirements are provided in Section V. Finally we provide the required statements about the approach in Section VI.

A. The bCMS Case Study

In this paper we use the SysML/KAOS language to model the bCMS case study. Here is an excerpt of the case study.

The bCMS is a distributed crash management system that is responsible for coordinating the communication between a FireStation Coordinator (FSC) and a Police Station Coordinator (PSC) to handle a crisis in a timely manner. Information regarding the crisis as it pertains to the tasks of the coordinators is updated and maintained during and after the crisis. There are two collaborative sub-systems. Thus, the global coordination is the result of the parallel composition of the (software) coordination processes controlled by the two (human) distributed coordinators. There is no central database; fire and police stations maintain separate databases and may only access information from the other database through the bCMS system. Each coordination process is hence in charge of adding and updating information in its respective database. Figure 1 shows the overall view of the bCMS case study.

Available at http://cseng0.site.uottawa.ca/cma2013re/CaseStudy.pdf
B. KAOS

The KAOS method provides four complementary sub-models that describe the system and its environment: a goal model, a responsibility model, an operation model and an object model [10]. In this paper, we mainly focus on the goal model. The main concept is the concept of goal. A goal is defined as a "prescriptive statement of intent the system should satisfy through cooperation of its agents" [10].

A goal model is an AND/OR graph where higher-level goals can be refined into lower-level sub-goals, and then, recursively, into low-level sub-goals that lead to the satisfaction of requirements of the system-to-be. The refinement relationship between a high level goal and its sub-goals is an AND/OR meta-relationship. When a goal is AND-refined into sub-goals, all of them must be satisfied for the parent goal to be satisfied. When a goal is OR-refined, the satisfaction of one of them is sufficient for the satisfaction of the parent goal. A goal that cannot be refined further is assignnable to an agent. An agent is an active system component having to play some role in goal satisfaction. A goal assignnable to an agent is a requisite. A requisite that is placed under the responsibility of an agent of the system-to-be is a requirement, whereas a requisite that is placed under the responsibility of an agent in the environment of the system is an expectation.

C. The NFR framework

The NFR framework [9] is a goal-oriented approach for modeling and analyzing non-functional requirements (NFRs). It provides a process model including the following main activities: capturing NFRs, identifying possible design alternatives for meeting NFRs (operationalization), dealing with interdependencies among NFRs, selecting alternatives and supporting decisions. In this framework, non-functional requirements are treated as Softgoal whose interdependencies are captured in graphs, the Softgoal Interdependency Graphs (SIGs). Softgoals are goals that need not be satisfied absolutely or can be partially satisfied. The SIG records the developers consideration of softgoal, and shows the interdependency among them. The impact of decisions is propagated through the graph to determine how well a chosen target system satisfies its NFR.

The framework also supports capturing design knowledge into the three following type of catalogues. NFR Type catalogues that include concepts about particular types of NFRs, such as security. Method catalogues, which capture knowledge that support softgoal refinement and operationalization. Correlation rule catalogues capture knowledge that helps in detecting implicit interdependencies among softgoals.
**D. SysML**

SysML (System Modelling Language) [5] is a general purpose modeling language for systems engineering applications, based on UML. While UML is very well suited to software engineering, it lacks elements for systems engineering such as requirements modeling. SysML is an extension of UML 2 and the part of UML which is reused by SysML is called UML4SysML. SysML main objective is to overcome the limitations of UML in the context of systems engineering. These limitations include, for example, the need to describe requirements directly in a specific model, and to ensure traceability to architecture, the need to represent strong links between software and hardware. The requirements diagram captures requirements hierarchies and requirements derivation. For example, the <<satisfy>> and <<verifiedBy>> relationships allow a modeler to relate a requirement to a model element. The requirements diagram provides a bridge between typical requirements management tools and system models. The readers are referred to [2] for the bCMS requirements modeling using SysML.

**II. THE APPROACH**

Figure 2 presents an overview of the SysML/KAOS meta-model. The proposed extensions are represented by white boxes, while the grey boxes are part of the SysML metamodel. Its instantiation allows us to obtain a hierarchy of requirements in the form of goals. Functional and non-functional requirements are specified at the same level of abstraction, but in two separate goal models. A final integrated goal model is then built. The latter describes the impact of non-functional requirements on functional ones. Indeed, non-functional requirements may have an impact on the choices and decisions taken when refining functional requirements. It can lead to the identification of new functional requirements that must be integrated with the initial functional goal model.
Goals are organized in refinement hierarchies. A requirement depicted by a goal (functional or non-functional) is either an abstract goal or an elementary goal. An abstract goal may be refined by several combinations of sub-goals. The relationship between an abstract goal and its sub-goals is a And/Or meta-relationship. A functional goal that cannot be further refined is a requisite and a non-functional goal that cannot be further refined is an elementary NFG. A requisite can be either an elementary requirement or an expectation. The first one is under the responsibility of an agent in the system, whereas the second one is under the responsibility of an agent in the environment of the system. The concept of contribution expresses possible solutions to satisfy elementary non-functional goals. A contribution can positively contribute to elementary non-functional goals and negatively to other ones. A conflict can appear between two non-functional goals when a contribution contributes positively to one of them and negatively to the other one. A contribution can be explicitly stated (explicit contribution) and considered to be helpful or induced from another one (induced contribution). It is important to note that a contribution must help at least once positively to the satisfaction of a non-functional goal. If this is not the case, its presence in the model cannot be justified.

The purpose of SYSML/KAOS is to consider the impact of non-functional goals on functional ones. This concept is represented in the meta-model as an association class between a contribution and a functional goal. The impactType property may have two possible values: positive or negative. A positive impact means that the contribution may positively impact the achievement of the related functional goal. In other words, the choice of this contribution can guarantee, with an acceptable level of confidence, that the target system perform the corresponding function with the expected quality. A negative impact means that the contribution may have a negative impact on the achievement of the related functional goal. The choice of this contribution does not guarantee that the target system will provide the service with all the qualities expected. In some case, it will be necessary to find the best compromise. The impactArgument property is intended to capture some arguments or rationales. In some cases, a contribution can lead to the identification of new functional goals that must be integrated with the initial functional goal model.

The approach is supported by a tool, implemented in the TOPCASED platform. It allows model creation from the extended meta-model.

III. BCMS REQUIREMENTS MODEL

The requirements were shared and numbered in a shared document. The list of the requirements we have taken into account is provided in Appendix V.

2Available at http://goo.gl/usjP5
By applying the SysML/KAOS approach on the bCMS case study, we first elaborate the functional goal hierarchy described in Fig. 3. Purple diamonds represent abstract goals; yellow diamonds represent expectations and green hexagons agents who are responsible for expectations. The circles with ‘X’ represent “AND refinement” relationships and solid lines with “<<OR>>” represent “OR refinement” relationships. The requirements listed in Section V are analyzed to extract functional goals. Note that some of them correspond to both functional and non-functional goals. The root of this hierarchy (Goal “To handle a crisis situation”) comes from Requirement R2. It is decomposed into four sub-goals, thanks to the AND relationship. The goals “To collect crisis related information”, “To define a crisis strategy” and “To affect resources to the crisis location” correspond exactly to Requirements R15, R13 and R11, whereas the goal “To supervise the crisis” is added to abstract the goals “To transmit crisis information” (Requirement R16) and “To maintain a control over the crisis” (Requirements R1 and R3). We also note that the description of R1 and R3 introduces elements related to the architectural aspect of the system such as communications between FSC, PSC and Firemen. Thus these elements are not considered in the goal model. The sub-goal “To collect crisis related information” is then AND-refined into three sub-goals: “To collect information from staff”, “To collect information from crisis location” and “To collect information from coordinators”. The goal “To collect information from crisis location” is refined thanks to the OR relationship into two sub-goals: “To collect information from victim” and “To collect information from witness”. These two goals are called “expectation” because they are placed under the responsibility of agents (“Victim” and “Witness”) in the environment of the system. Similarly, the other sub-goals of the root of the hierarchy are recursively decomposed into sub-goals (Fig. 3).

Fig. 4 shows the non-functional goal model. Purple diamonds represent abstract goals, blue diamonds represent elementary goals and yellow diamonds represent contributions. Solid lines relate explicit contributions to elementary goals and dotted lines relate induced contributions to elementary goals. The type (positive or negative) of the contribution is represented by a “+” or “−” sign on the link. We can note that Requirement R2 gives a functional goal but also the non-functional goal “Good performance [System]”. The goal “Response time [System]” comes from several requirements (R4, R5, R6, R7). A conflict can appear between two non-functional goals when a contribution contributes positively to one of them and negatively to the other one. For instance “The system checks the validity of received information” contributes positively to “Correct Reception [Crisis Detail]” and negatively to “Response time [System]”. A compromise needs to be found.

The last step of the SysML/KAOS approach is to analyze and describe the impact of the non-functional goal model on the functional goal model. New functional goals can be added to satisfy non-functional goals. We obtain as a result, a new functional goal model called the integrated goal model. For example, the contribution “The system checks the validity of received info” and those associated with the security of the crisis information (see Fig. 4) can have an impact on the functional goal “To collect crisis related information”. Thus, in Fig. 5, the goals “To check the validity of collected information” and “To secure collected information” are added as sub-goals of the goal “To collect crisis related information”. The first is an elementary goal placed under the responsibility of the system agent “Validity checking method” the second is OR-refined into two elementary sub-goals: “Use noise addition” and “Use encryption technique”.

![Fig. 5. Extract of the integrated goal model](image-url)
IV. CONCLUSION

In this paper, we have described the requirements modeling of the bCMS case study using the SysML/KAOS approach. The approach allows representing non-functional requirements at the same level of abstraction as functional requirements, but in separate models. Then the impact of non functional requirements on functional requirements are captured and specified in an integrated final goal model. Note that an interesting feature of SYSML, not presented in this paper, is the possibility of linking requirements models with specification and design models through the definition of traceability links.

REFERENCES


APPENDICES

V. REQUIREMENTS STUDIED

<table>
<thead>
<tr>
<th>Req. ID</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>A FSC maintains control over a crisis situation by communicating with the police station coordinator (PSC) as well as firemen.</td>
</tr>
<tr>
<td>R2</td>
<td>To resolve a crisis situation as quickly and cost-effectively as possible in cooperation with other coordinator.</td>
</tr>
</tbody>
</table>

VI. STATEMENTS

Classification of the approach

In this paper, only the goal-oriented feature of SysML/KAOS is used.

Software development phases

As SysML/KAOS is an extension of SysML, it covers basically all the phases.

What is modelled
The system in its environment (links between requirements, environment, and systems parts).

*The modeling approach*

See Section II.

*References*

For more on SYSML/KAOS see [8], [7], [6].

*Relation between sub-models*

See Section III.