Self-Adaptive Systems Requirements Modelling: Four Related Approaches Comparison

M. Ahmad*, João Araújo†, Nicolas Belloir‡, Jean-Michel Bruel*,
Christophe Gnaho§, Régine Laleau§, and Farida Semmak§
*Université de Toulouse, France
{ahmad,bruel}@irit.fr
†CITI, FCT/Universidade Nova de Lisboa, Portugal
joao.araujo@fct.unl.pt
‡LIUPPA, France
nicolas.belloir@univ-pau.fr
§Université Paris-Est Créteil, France
laleau@u-pec.fr, christophe.gnaho@free.fr, semmak.farida@wanadoo.fr

***Abstract*—When developing Self Adaptive Systems (SAS), their highly adaptiveness has to be taken into account as early as the requirements elicitation. Because such systems modify their behaviour at run-time in response to changing environmental conditions, Non Functional Requirements (NFR's) play an important role. One has to identify as early as possible the requirements that are adaptable. Because of the inherent uncertainty in these systems, goal based approaches can help in the development of their requirements. In order to cope with this purpose, we have defined a combined approach based on several requirements modelling techniques. In this paper we use a common case study and well defined comparison criteria to illustrate the way those techniques can benefit from each other. This submission is a synthesis and hence make some reference of more specific requirements models submissions.**

I. Introduction

This paper aims at studying the differences and potential combination of 4 different requirements' modelling techniques: KAOS [8], a goal-based approach (detailed in section II-B), SYSML [5], a general purpose system modelling notation (detailed in section II-C), SYSML/KAOS a combination of KAOS and SYSML (detailed in section II-D) and RELAX [6] a dedicated language for adaptive systems. We have taken the opportunity of the CMA@RE workshop in order to compare those techniques that were chosen to provide the best approach to deal with the specificities of Self-Adaptive Systems requirements.

In section II we describe the context in which this study has been conducted, as well as the minimum of background needed to understand the models; in section III we provide the different requirements models we have developed for the bCMS case study; in section IV we provide our inputs to the defined comparison criteria; and we analyse the results of the parallel efforts; and finally in section V we conclude this study.

II. Context

In order to contribute to the third International Comparing *Requirements* Modelling Approaches (CMA@RE) workshop, we have selected different requirements modelling approaches to discuss and evaluate their differences and mutual benefits.

We have only focused on the requirements part of the case study[1] but have fulfilled the matching comparison criteria[2].

The modelling have been made separately and without communication between 4 different groups of people.

1) João Araújo has long worked around KAOS [17] and has made a pure KAOS modelling of the requirements;
2) Nicolas Belloir did the SYSML modelling;
3) Régine Laleau, Christophe Gnaho and Farida Semmak are the initiator of the SYSML/KAOS approach and hence did this part of the modelling;
4) Manzoor Ahmad is currently finishing his Ph.D. on the combination of RELAX/SYSML/KAOS and did what is simply called the RELAX modelling in the remaining of this paper. RELAX is originally a language defined by Jon Whittle, Pete Sawyer, Nelly Bencomo, Betty H.C. Cheng, and Jean-Michel Bruel [6].

A. Case Study Description

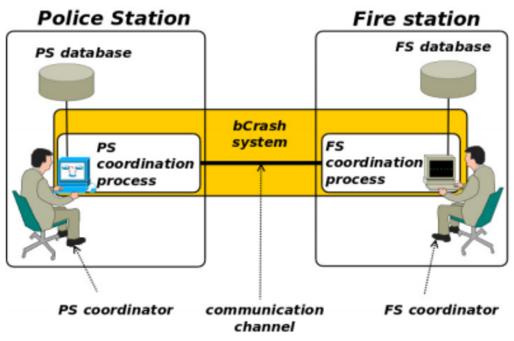


Fig. 1. Overall view of the environment and the desired system

The bCMS system is a distributed crash management system that is responsible for coordinating the communication

[1]Available at http://cserg0.site.uottawa.ca/cma2013re/CaseStudy.pdf.

[2]Available at http://cserg0.site.uottawa.ca/cma2013re/ComparisonCriteria.pdf.

between a fire station coordinator (FSC) and a police station coordinator (PSC) to handle a crisis in a timely manner (see Fig. 1). Internal communication among the police personnel (including the PSC) is outside the scope of the desired system. The same assumption applies to the fire personnel (including the FSC). Information regarding the crisis as it pertains to the tasks of the coordinators will be updated and maintained during and after the crisis. For the full description of the case study, see http://cserg0.site.uottawa.ca/cma2013re/CaseStudy.pdf.

B. KAOS

KAOS (Knowledge Acquisition in autOmated Specification) [8] is a GORE (Goal Oriented Requirements Engineering) framework whose emphasis is on semi-formal and formal reasoning about behavioral goals to derive goal refinements, operationalizations, conflict management and risk analysis. In KAOS goals can be refined into subgoals through and/or decompositions. Goals can also be refined into requirements (i.e., a goal whose responsibility is assigned to a software agent), or expectations (i.e., a goal whose responsibility is assigned to an environment agent). KAOS also introduces the concept of obstacle as a situation that prevents the achievement of a goal [9]. The resolution to the obstacle is expressed in the form of a goal that can also be refined into requirements or expectations. The main steps for building KAOS specifications from high level goals are:

1) goals development: goals refinement through the identification of new and more specific goals that characterize the high-level ones;
2) objects identification objects identification in the formulation of the goal, definition of the links among them, and description of the domain properties;
3) operations identification: identification of object state transitions that are significant to the goal;
4) goals operationalization: specification of operations in order to satisfying all goals;
5) responsibilities assignment: mapping of agents to leaf goals and operations assignment to agents. Here we give emphasis on the goals development. We used the Objectiver tool http://www.objectiver.com/.

C. SysML

SYSML (System Modelling Language) is a general purpose modelling language for systems engineering applications. It supports the specification, analysis, design, verification and validation of a broad range of systems and systems-of-systems. These systems may include hardware, software, information, processes, personnel, and facilities. It includes a graphical construct to represent text based requirements and relate them to other model elements.

SYSML was created in 2001. Its first official OMG specification has been released in September 2007. The current version is 1.3 [5]. Its initiators were the Object Management Group (OMG) and the International Council on Systems Engineering (INCOSE). Its main authors were Conrad Bock, Cris Kobryn and Sanford Friedenthal.

SYSML provides graphical representations with a semantic foundation for modelling system requirements, behavior, structure, and parametrics. The requirements diagram captures requirements hierarchies and requirements derivation (e.g., see Fig. 3), and the `<<satisfy>>` and `<<verify>>` relationships allow a modeller to relate a requirement to a model element, e.g. `<<block>>`, that satisfies or verifies the requirements. The requirement diagram provides a bridge between typical requirements management tools and system models.

D. SysML/Kaos

The SYSML/KAOS language is an extension of the SYSML requirements language [5] with the most relevant concepts of the KAOS goal model [8] and NFR Framework [7], two goal based approaches largely recognized and used in requirements engineering over the past decade. The main idea is to introduce the concept of goal in a SYSML requirements model, which would help to take advantage of the contribution of SYSML while providing a clear definition of requirements and relationships between them. For that, we combine the concepts of KAOS that are better suited to represent functional requirements with concepts of the NFR model which are most relevant to specify non-functional requirements. This allows the integration of non-functional requirements much earlier and at the same level of abstraction than functional requirements. As SYSML/KAOS is a goal based approach, it allows detection and resolution of conflicts among requirements and it helps reasoning about alternative configurations. In addition, it specifically allows the detection and analysis of impact of non-functional requirements on functional requirements. 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. The instantiation of the SYSML/KAOS metamodel allows us to obtain a hierarchy of requirements in the form of goals. In a first step, 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 reflects the impact of non-functional requirements on functional ones.

E. RELAX

RELAX [6] is a requirements engineering language for Dynamic Adaptive Systems (DAS), where explicit constructs are included to handle uncertainty. RELAX takes the form of structured natural language, including operators designed specifically to capture uncertainty [14], their semantics is also defined. Uncertainty can be environmental and behavioural; environmental uncertainty is due to changing environmental conditions such as sensor failure, noisy networks, malicious threats and unexpected human input. Here uncertainty refers to maintaining the same requirements in unknown contexts.

Behavioral uncertainty refers to situations where requirements themselves need to change. For example, the requirements of a space probe may change mid-flight in order to pursue science opportunities not foreseen by the designers. It is difficult to know all requirements changes at design time and, in particular, it may not be possible to enumerate all possible alternatives. Behavioral uncertainty refers to the need to change system behavior at run time in response to the environmental uncertainty.

The RELAX vocabulary helps in relaxing requirements when environment changes so it enables the analysts to identify the point of flexibility in their requirements. For this purpose RELAX classifies requirements into two types: variant or RELAX-ed requirements that can be RELAX-ed when the environment changes, and invariant requirements that are fixed and cannot be changed since they represent the main functionality of the system. In RELAX the conventional modal verb SHALL is retained and RELAX operators (e.g. *AS CLOSE AS POSSIBLE TO*, *AS EARLY AS POSSIBLE* etc., [6]) are introduced to provide more flexibility in how and when that functionality may be delivered. More specifically, for requirements that are left partially unsatisfied, the introduction of an alternative, temporal or ordinal RELAX-ation modifier will define the requirement as RELAX-able. These operators define constraints on how a requirement can be RELAX-ed at run-time. In addition, it is important to indicate what uncertainty factors warrant a RELAX-ation of these requirements, thereby requiring adaptive behavior. This information is specified using the MON (monitor), ENV (environment), REL (relationship) and DEP (dependency) keywords. SYSML incorporates requirements through requirements diagram. We have developed a Domain Specific Language (DSL) which links SYSML and RELAX [15]. SYSML provides a development environment and a graphical support for expressing all the variables of RELAX and helps in bridging the gap between requirements and the overall system model.

III. MODELLING OF THE REQUIREMENTS

We have chosen an (illustrative) subset of the bCMS requirements. The requirements were shared and numbered in a shared document[3] (see Fig. 2). The formal numbering of the requirements has been used to make sure we could compare the way each of them have been modelled by each technique.

A. SysML

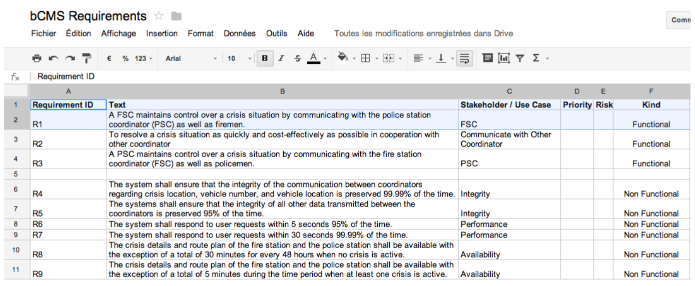
SYSML has a dedicated diagram to show the requirements and more specifically their relationship. This requirement diagram can be generated automatically from requirements tools such as DOORS[4]. In the context of the case study, we have used the TOPCASED[5] free tool. Due to page limitation we only show an extract of the SYSML model (see [2] for more details).

Fig. 3 shows some of the functional requirements.

[3]Available at http://goo.gl/uscP5.

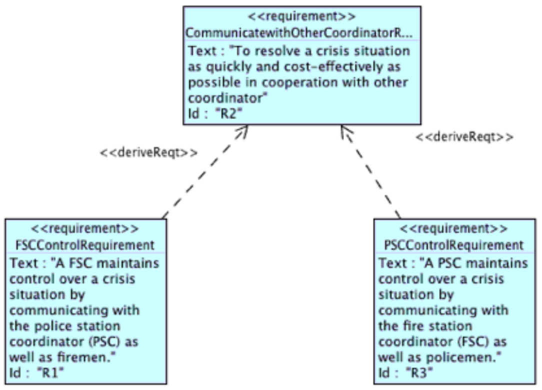
[4]http://www-03.ibm.com/software/products/us/en/ratidoorfami/

[5]Available at http://www.topcased.org/



	A	B	C	D	E	F
1	Requirement ID	Text	Stakeholder / Use Case	Priority	Risk	Kind
2	R1	A FSC maintains control over a crisis situation by communicating with the police station coordinator (PSC) as well as firemen.	FSC			Functional
3	R2	To resolve a crisis situation as quickly and cost-effectively as possible in cooperation with other coordinator	Communicate with Other Coordinator			Functional
4	R3	A PSC maintains control over a crisis situation by communicating with the fire station coordinator (FSC) as well as policemen.	PSC			Functional
5						
6	R4	The system shall ensure that the integrity of the communication between coordinators regarding crisis location, vehicle number, and vehicle location is preserved 99.99% of the time.	Integrity			Non Functional
7	R5	The systems shall ensure that the integrity of all other data transmitted between the coordinators is preserved 95% of the time.	Integrity			Non Functional
8	R6	The system shall respond to user requests within 5 seconds 95% of the time.	Performance			Non Functional
9	R7	The system shall respond to user requests within 30 seconds 99.99% of the time.	Performance			Non Functional
10	R8	The crisis details and route plan of the fire station and the police station shall be available with the exception of a total of 30 minutes for every 48 hours when no crisis is active.	Availability			Non Functional
11	R9	The crisis details and route plan of the fire station and the police station shall be available with the exception of a total of 5 minutes during the time period when at least one crisis is active.	Availability			Non Functional

Fig. 2. Shared requirements document



Text : "To resolve a crisis situation as quickly and cost-effectively as possible in cooperation with other coordinator"
Id : "R2"

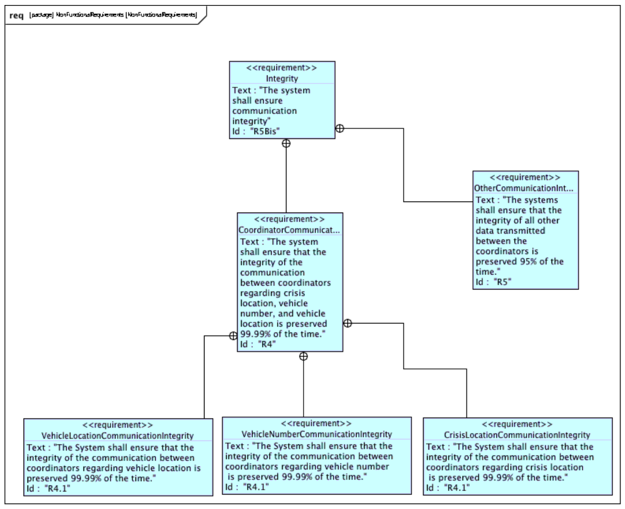
Text : "A FSC maintains control over a crisis situation by communicating with the police station coordinator (PSC) as well as firemen."
Id : "R1"

Text : "A PSC maintains control over a crisis situation by communicating with the fire station coordinator (FSC) as well as policemen."
Id : "R3"

Fig. 3. Functional requirements in SysML

The <<deriveReqt>> dependency relationship is used to show that a requirement "comes from" another.

Fig. 4 shows non-functional ones. The composition relationship illustrates the fact that a requirement is decomposed into several ones.


req [package] NonFunctionalRequirements [NonFunctionalRequirements]
<<requirement>>
Integrity
<<requirement>>
OtherCommunicationInt...
<<requirement>>
CoordinatorCommunicat...
<<requirement>>
VehicleLocationCommunicationIntegrity
<<requirement>>
VehicleNumberCommunicationIntegrity
<<requirement>>
CrisisLocationCommunicationIntegrity

Text : "The system shall ensure communication integrity"
Id : "R5Bis"

Text : "The systems shall ensure that the integrity of all other data transmitted between the coordinators is preserved 95% of the time."
Id : "R5"

Text : "The system shall ensure that the integrity of the communication between coordinators regarding crisis location, vehicle number, and vehicle location is preserved 99.99% of the time."
Id : "R4"

Text : "The System shall ensure that the integrity of the communication between coordinators regarding vehicle location is preserved 99.99% of the time."
Id : "R4.1"

Text : "The System shall ensure that the integrity of the communication between coordinators regarding vehicle number is preserved 99.99% of the time."
Id : "R4.1"

Text : "The System shall ensure that the integrity of the communication between coordinators regarding crisis location is preserved 99.99% of the time."
Id : "R4.1"

Fig. 4. Non-functional requirements in SysML

B. KAOS

We have chosen to model the system as a single unit and put the emphasis on the communication concern to resolve the crises both at function and non-functional levels. Communication is satisfied by establishing communication, exchanging crisis details, coordinating route plan, communicating actions (e.g. dispatch, arrival). These are expressed as goals (blue parallelograms). Obstacles (red parallelograms)can prevent some goals to be satisfied. In this example, we have three obstacles: timeout, communication not available and unreachability of destination. These obstacles should be resolved. For example, to resolve timeout, an obstacle to the goal coordinate route plan, an alert should be given, or a report should be sent.

The NF concerns which we focus on are *multiplicity*, *performance*, *confidentiality* and *availability* (see Fig. 5). In the model we specify conflicts (the lightning symbol) between multiplicity and availability, and between performance and multiplicity and confidentiality.

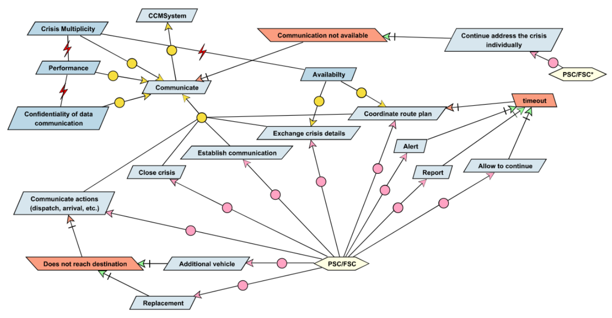


Fig. 5. KAOS model

The model is a high level one that must be refined to show the *operationalisation* of goals and softgoals. Due to page limitation we only show an extract of the KAOS model (see [4] for more details).

C. SysML/Kaos

In SYSML/KAOS functional and non-functional requirements can be specified in two separate goal models. A final integrated goal model is then built. It 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. Due to page limitation we only show an extract of the SYSML/KAOS model (see [3] for more details).

Fig. 6 shows the non-functional goal model. Some of the concepts are taken from the NFR method [7]. As functional goals, non-functional goals can be refined into subgoals thanks to the AND/OR refinement mechanism. The concept of *contribution* (yellow diamonds in Fig. 6) 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.

The specificity of SYSML/KAOS is to consider the impact of non-functional goals on functional ones. More precisely, a

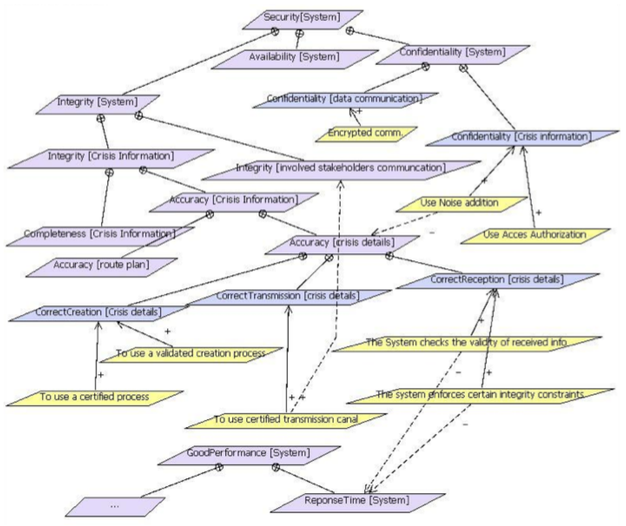

Security[System]
Availability [System]
Confidentiality [System]
Integrity [System]
Confidentiality [data communication]
+
Encrypted comm.
Confidentiality [Crisis information]
Integrity [Crisis Information]
Integrity [involved stakeholders communcation]
+
Use Noise addition
+
Accuracy [Crisis Information]
Completeness [Crisis Information]
-
Use Acces Authorization
Accuracy [crisis details]
Accuracy [route plan]
CorrectReception [crisis details]
CorrectTransmission [crisis details]
CorrectCreation [Crisis details]
+
To use a validated creation process
+
The System checks the validity of received info
+
-
+
To use a certified process
+ +
The system enforces certain integrity constraints
To use certified transmission canal
-
GoodPerformance [System]
...
ReponseTime [System]

Fig. 6. Non-functional goal model

contribution can lead to the identification of new functional goals. For example, in Fig. 7 that shows an extract of the integrated goal model, blue diamonds are new functional goals linked to the relevant contributions (blue hexagons).

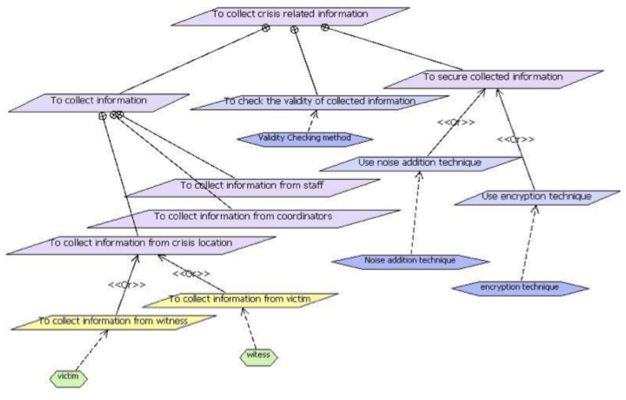

To collect crisis related information
To secure collected information
To collect information
To check the validity of collected information
<<Or>>
Validity Checking method
<<Or>>
Use noise addition technique
To collect information from staff
Use encryption technique
To collect information from coordinators
To collect information from crisis location
Noise addition technique
<<Or>>
<<Or>>
encryption technique
To collect information from victim
To collect information from witness
witess
victim

Fig. 7. Extract of the integrated goal model

D. RELAX/SysML/KAOS

We have first applied the RELAX process on bCMS requirements to get invariant and RELAX-ed requirements. For RELAX-ed requirements, all the uncertainty factors were identified. Then using the correlation in Table II we have modelled the bCMS system requirements with the SYSML/KAOS approach.

Following are some extracts of the integrated models, due to page limitation (see [1] for more details). The invariant and RELAX-ed requirements that were identified:

- Invariant requirements: R1, R2, R3.
- Relax-ed Requirements: R4, R8.

Here is for example the RELAX-ed version of R4 (*The system shall ensure that the integrity of the communication between coordinators regarding crisis location, vehicle number,*

TABLE I
RELAX-ED REQUIREMENT UNCERTAINTY FACTORS

Uncertainty Factors	Details
ENV	integrity of the communication between coordinators, the authenticity of the coordinators to avoid the communication compromiser
MON	use secure communication channel, use PIN code, use additional information, communication compromiser
REL	secure communication channel ensures the integrity of the communication between coordinators, PIN code and Additional information ensures that the authenticity of the coordinator is in place, The communication compromiser compromises the integrity of coordinators

and vehicle location is preserved AS CLOSE AS POSSIBLE TO 99.99% of the time. see Table I):

The SYSML/KAOS approach helps in modelling the impact of an NFG on an FG. In Fig. 8, the abstract functional goal *"A Fire Station Coordinator (FSC) maintains control over a crisis situation by communicating with the Police Station Coordinator (PSC) as well as firemen"* is refined into three sub goals: (i) *To get resources to the crisis location*, (ii) *To handle crisis related information* and (iii) *To provide executable instructions to staff*. The contribution goal *Fire Station Coordinator* has a direct and positive impact on each of the three functional sub goals.

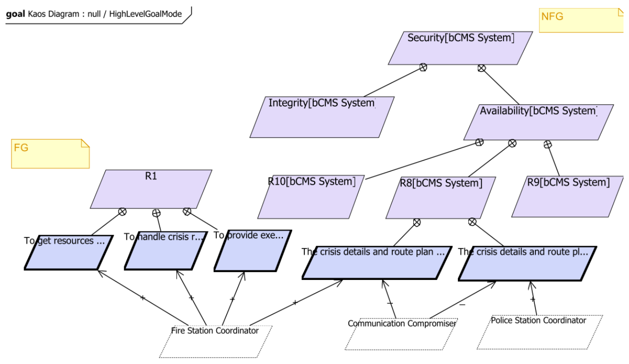


Fig. 8. High level goal model

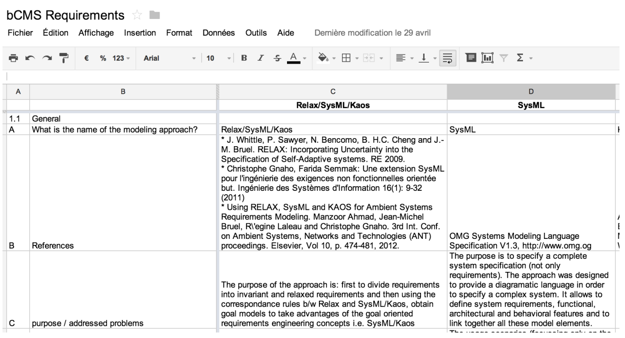
IV. COMPARISON CRITERIAS

The criterias have been filled using the provided pdf file[6], and then summarized in a shared document[7] (see Fig. 9).

As one would have expected the integrated approaches (SYSMLKAOS and RELAX/SYSML/KAOS) have better criteria coverage than each individual one as they integrate the best part of each of their components. The way we compiled the criteria (through a table) helped us highlighting the complementarity of very different approaches such as SYSML (which can be seen as a broad notation) and KAOS (being focused mainly on the requirements phase).

[6]Available at http://cserg0.site.uottawa.ca/cma2013re/ComparisonCriteria.pdf.

[7]Available at http://goo.gl/uscP5.


bCMS Requirements
Fichier Édition Affichage Insertion Format Données Outils Aide Dernière modification le 29 avril
€ % 123 Arial 10 B I S A

A	B	C	D
		Relax/SysML/Kaos	**SysML**
1.1	General		
A	What is the name of the modeling approach?	Relax/SysML/Kaos	SysML
B	References	* J. Whittle, P. Sawyer, N. Bencomo, B. H.C. Cheng and J.-M. Bruel. RELAX: Incorporating Uncertainty into the Specification of Self-Adaptive systems. RE 2009. * Christophe Gnaho, Farida Semmak: Une extension SysML pour l'ingénierie des exigences non fonctionnelles orientée but. Ingénierie des Systèmes d'Information 16(1): 9-32 (2011) * Using RELAX, SysML and KAOS for Ambient Systems Requirements Modeling. Manzoor Ahmad, Jean-Michel Bruel, R\'egine Laleau and Christophe Gnaho. 3rd Int. Conf. on Ambient Systems, Networks and Technologies (ANT) proceedings. Elsevier, Vol 10, p. 474-481, 2012.	OMG Systems Modeling Language Specification V1.3, http://www.omg.og
C	purpose / addressed problems	The purpose of the approach is: first to divide requirements into invariant and relaxed requirements and then using the correspondance rules b/w Relax and SysML/Kaos, obtain goal models to take advantages of the goal oriented requirements engineering concepts i.e. SysML/Kaos	The purpose is to specify a complete system specification (not only requirements). The approach was designed to provide a diagramatic language in order to specify a complex system. It allows to define system requirements, functional, architectural and behavioral features and to link together all these model elements.

Fig. 9. Comparison criteria for the 4 approaches

We have identified several lacks in the criteria we would like to raise and which would hopefully bring some discussions during the workshop:

A. Too Much Emphasis on Composition

Indeed one of the main goal of the comparison criteria was to highlight *composable* elements and composition mechanisms. Working on approaches closer to (mainly graphical) notation rather than textual notation such as programming languages was not easy for us. In addition, the composition concept for requirements models needs to be precisely defined as well as its link with the refinement concept.

B. Lack of Dynamic Aspects

The possibility to express dynamic models have not been used in our modelling of the case study but this is mainly because we found that this part would have little impact on the filling of the comparison criteria. In our modelling of highly dynamic systems our approach (especially the integrated ones) would have benefit of criteria that would address links between system and its environment or requirements with the system itself (e.g. satisfiability).

C. Lack of Emphasis on Traceability

In the same idea, we find very important the traceability support of modelling techniques (may be as much as composition mechanisms). Some comparison criteria should address more specifically this concern.

D. Lack of Emphasis on Human Aspects

It is surprising that almost no criteria addresses the place of the humans (e.g. stakeholders, modellers, architects). As a consequence, poor methodological and process concerns are addressed by the questions, in our humble opinion.

E. Analysis and comparison

Nevertheless, thanks to the comparisons criteria, we have been able to improve the correlation between important concepts. We have been able to draw links between them. In Table II, we have shown how several key concepts are taken into account in the selected models. Most of the time, the concepts are not fully covered (e.g. `<<satisfy>>` for monitoring in SYSML, this stereotype is used between a block and a requirement), but we have indicated in the Table II the closest mechanism that supports the concepts. In SYSML/KAOS, requirements are described in the form of goals, SYSML describes requirements in textual form while RELAX requirements are also in textual form with an enhanced version i.e. requirements divided into invariant and RELAX-ed requirements with uncertainty factors added to it. SYSML/KAOS has the concepts of AND/OR relationship and Contribution. Contribution describes the characteristics of the contribution and it provides two properties: ContributionNature and ContributionType. The first one specifies whether the contribution is positive or negative, whereas the second one specifies whether the contribution is direct or indirect, SYSML has `<<verify>>` and `<<refine>>` relationships while for RELAX, we have REL variable which identifies the relationship between ENV and MON. For Dependency/Impact, SYSML/KAOS describes it as an Impact of an NFG on Functional Goal (FG); this Impact can be positive or negative. In SYSML/KAOS meta-model (see [11]), the Impact is an Association Class between Contribution Goal and an FG. It captures the fact that a Contribution Goal has an impact on an FG which in turn shows it as an Impact of an NFG on an FG. while for SYSML, we have the concept of `<<derive>>` which shows the dependency between requirements, RELAX has positive and negative dependency. To deal with monitoring, SYSML/KAOS has the Contribution goal concept which is used to satisfy an NFG, SYSML has `<<satisfy>>` which is used when a `<<block>>` satisfies a `<<requirement>>` while for RELAX, we have the concept of MON which is used to measure the environment i.e. ENV. SYSML/KAOS has a tool called SYSML/KAOS editor, SYSML has a number of tools e.g. eclipse (see http://www.eclipse.org/), papyrus (see http://www.papyrusuml.org), topcased (see http://www.topcased.org/) etc. and for RELAX, we have eclipse based COOL RELAX editor.

TABLE II
CORRELATION B/W RELAX SYSML/KAOS

Concepts/Approaches	**SysML/KAOS**	**SysML**	**RELAX**
Requirements Description	AbstractGoal ElementaryGoal	Textual Requirements	Relaxed Requirement ENV
Monitoring	Contribution Goal	<<satisfy>>	MON
Relationship	AND/OR, Contribution (Contribution Nature: Positive Negative Contribution Type: Direct (Explicit) Indirect (Implicit))	<<verify>> <<refine>>	REL
Dependency/Impact	Contribution Nature: Positive Negative	<<derive>> <<contain>>	DEP: Positive Negative
Tools	Eclipse based SysML/KAOS Editor	Eclipse/Papyrus/Topcased/ …..	Eclipse based COOL RELAX editor

In the above table, the KAOS concepts are already integrated in the SYSML/KAOS approach. In terms of the comparison criteria, we have experienced that the key modelling concepts are more suitable at the software level than the requirements level.

V. CONCLUSION

In this paper we have presented the same requirements modelled through several different but related modelling techniques. The proposed RELAX/SYSMLKAOS approach allows to model the non-functional requirements of self adaptive systems.

We have mainly focused our efforts in the readability of the models, the usability of our inputs for the workshop (shared documents, tables and usable data, etc.).

REFERENCES

[1] M. Ahmad and J.-M. Bruel, *bCMS requirements modelling using RELAX/SysML/KAOS*, submitted to the 3rd CMA workshop at RE'2013.
[2] N. Belloir and J.-M. Bruel, *bCMS requirements modelling using SysML*, submitted to the 3rd CMA workshop at RE'2013.
[3] R. Laleau, C. Gnaho, F. Semmak and J.-M. Bruel, *bCMS requirements modelling using SysML/KAOS*, submitted to the 3rd CMA workshop at RE'2013.
[4] J. Araújo and J.-M. Bruel, *bCMS requirements modelling using KAOS*, submitted to the 3rd CMA workshop at RE'2013.
[5] OMG. SysML specification. v1.3, 12/06/2012. Available at http://www.sysml.org/docs/specs/OMGSysML-v1.3-12-06-02.pdf.
[6] Jon Whittle, Pete Sawyer, Nelly Bencomo, Betty H.C. Cheng, and Jean-Michel Bruel. RELAX*: Incorporating Uncertainty into the Specification of Self-Adaptive systems*, Proceedings of the 2009, 17th IEEE International Requirements Engineering Conference, Pages: 79-88.
[7] Lawrence Chung, John Mylopoulos, Eric S.K. Yu, Brian Nixon. *Non Functional Requirements in Software Engineering*, Kluwer, 1999.
[8] Axel Van Lamsweerde. *Requirements Engineering: From System Goals to UML Models to Software Specifications*, Wiley, 2009.
[9] Axel Van Lamsweerde and Emmanuel Letier. *Handling Obstacles in Goal-Oriented Requirements Engineering*, IEEE Trans. Software Engineering 2000.
[10] Annie I. Anton. *Goal based requirements analysis*, In Proceedings of Int. Conf. on Requirements Engineering, 1996.
[11] Christophe Gnaho, Farida Semmak. *Une extension SYSML pour l'ingénierie des exigences non-fonctionnelles orientée but*, Revue Ingénierie des Systèmes d'Information, Vol 16/1, 23 pages, 2011.
[12] Christophe Gnaho, Farida Semmak, Régine Laleau. *An overview of a SysML extension for goal-oriented NFR modelling*, Seventh IEEE International Conference on Research Challenges in Information Science, May 29-31 2013, Paris, France.
[13] Régine Laleau, Farida Semmak, Abderrahman Matoussi, Dorian Petit, Ahmed Hammad, Bruno Tatibouet. *A first attempt to combine SysML requirements diagrams and B*, Innovations in Systems and Software Engineering, Springer, Springer, 6(1-2): 47-54, 2010.
[14] Jon Whittle, Pete Sawyer, Nelly Bencomo and Betty H. C. Cheng. *A Language for Self-Adaptive System Requirements*, International Workshop on Service-Oriented Computing: Consequences for Engineering Requirements, 2008. SOCCER '08.
[15] Manzoor Ahmad. *First step towards a domain specific language for self-adaptive systems*, 10th Annual International Conference on New Technologies of Distributed Systems (NOTERE'10) IEEE, 285-290.
[16] Manzoor Ahmad, Jean-Michel Bruel, Régine Laleau and Christophe Gnaho. *Using RELAX, SysML and KAOS for Ambient Systems Requirements Modeling*, 3rd Int. Conf. on Ambient Systems, Networks and Technologies (ANT) proceedings. Elsevier, Vol 10, p. 474-481, 2012.
[17] P. Espada, M. Goulão, and J. Araújo, *Measuring complexity and completeness of KAOS goal models*, Proceedings of the International Workshop on Empirical Requirements Engineering (EmpiRE 2011), at the 19th International Requirements Engineering Conference (RE 2011): IEEE Computer Society, pp. 29-32, 08, 2011.