Towards Improved Requirements Engineering with SysML and the User Requirements Notation

Daniel Amyot, Amal Ahmed Anda, Malak Baslyman
University of Ottawa
Ottawa, Canada
{damyot, aanda027, mbasl071}@uottawa.ca

Lysanne Lessard
University of Ottawa and IRHM
Ottawa, Canada
lessard@telfer.uottawa.ca

Jean-Michel Bruel
IRIT and IUT de Blagnac
Toulouse, France
bruel@irit.fr

***Abstract*—The Systems Modeling Language (SysML) is a popular and standardized UML profile for systems engineering applications. In addition to seven conventional UML diagram types, SysML supports requirement diagrams and tables that can be used to capture requirements, their attributes, and their relationships. However, several important concepts such as goals and contributions are not predefined in SysML, hindering the reasoning about tradeoffs and adaptation, especially in emerging socio-cyber-physical systems such as smart cities. In this paper, we provide a preliminary investigation of different ways of combining SysML with the User Requirements Notation (URN), a standard that focuses on the modeling and analysis of goals and scenarios. We argue that SysML and URN are complementary and synergistic, and that their combination enables new requirements modeling, analysis, and management opportunities for new types of systems. Examples of potential integration approaches are discussed, briefly assessed, and illustrated with existing tools for SysML, URN, and requirements management.**

***Index Terms*—Goal modeling, requirements management, SysML, tool integration, User Requirements Notation.**

I. Introduction

The *Systems Modeling Language* (SysML) is a profile of the Unified Modeling Language (UML) standardized by the Object Management Group (OMG) in collaboration with the International Council on Systems Engineering (INCOSE). The latest version (1.4) is available since September 2015 [28]. Whereas UML focuses mainly on the modeling and analysis of software [29], SysML addresses systems that may mix hardware and software, such as *cyber-physical systems* (CPS), as well as *systems of systems* (SoS).

SysML is an extended subset of UML 2 that:

- reuses sequence, state machine, use case, and package diagrams;
- modifies activity diagrams, class diagrams (as block definition diagrams) and composite structure diagrams (as internal block diagrams); and
- adds two new types of diagrams, namely parametric diagrams and *requirement diagrams* (also visualizable in a tabular format).

The *block* element is the basic unit of structure in SysML and is used to represent various elements, including hardware, software, and personnel. From a requirements engineering perspective, SysML goes beyond UML and defines model elements for problems, rationales, stakeholders, and requirements. A SysML *requirement* has a name, an identifier, a text body, and possibly other user-defined attributes. Requirements can be linked to many types of model elements to enable traceability and analysis. Predefined *relationships* include requirement containment, copy, derivation, satisfaction (e.g., by use cases), verification (by test cases), refinement, and generic traceability.

A *requirement diagram* contains requirements and their relationships, and potentially other modeling elements (e.g., a block, a use case, or a test case) and relationships with requirements. See the bottom part of Fig. 1 for an example. These diagrams have shown benefits in the understanding of requirements, at least for small diagrams [32]. The same information can also be visualized using a *tabular format*, more amenable to the checking of relationships (e.g., as a traceability matrix). Such tabular views are usually more scalable than their graphical counterparts, especially when dealing with a large number of requirements and relationships.

Compared to UML, SysML is simpler as it has fewer types of diagrams, offers more precise concepts and semantics for systems modeling, and offers better support for requirements engineering activities. Yet, we believe that SysML misses important modeling concepts, notably *goals*. Among many applications, goal-oriented requirements engineering (GORE) enables the modeling of socio-technical aspects and non-functional requirements, traceability between requirements and stakeholder objectives, tradeoff analysis between conflicting goals, support for adaptive behavior, validation with management stakeholders, and holistic decision making [7][35][36].

Given the importance of goals in social considerations, one important motivation for combining goals with SysML is hence to provide a better requirements engineering foundation for existing *socio-cyber-physical systems* (SCPS), such as air traffic control systems, and emerging ones such as smart homes/cities [34], human-oriented services exploiting the Internet of Things (IoT) [39], adaptive SoS [12], and intelligent production networks [16].

Although there exist several goal modeling languages, only one is standardized at this time, namely the *Goal-oriented Requirement Language* (GRL), part of the User Requirements Notation (URN) [6][20].

2332-6441/16 $31.00 © 2016 IEEE
DOI 10.1109/RE.2016.58

RE 2016, Beijing, China
RE@Next! Paper

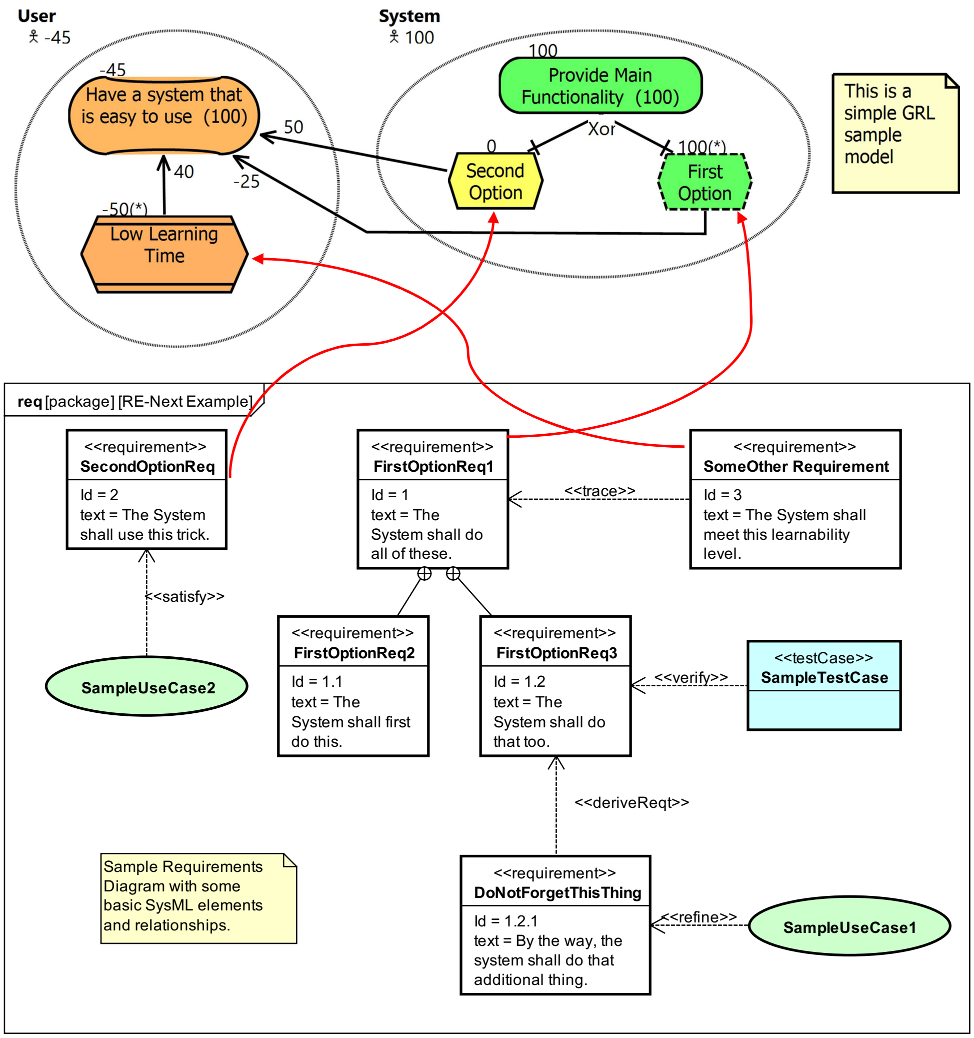

Fig. 1. Illustrative GRL model (top) and SysML requirement diagram (bottom), with several relationships

GRL supports concepts for goals, softgoals, tasks, resources, actors, and indicators, as well as several types of relationships (decomposition, contribution, and dependency) and execution contexts (strategies). The top part of Fig. 1 shows a sample GRL model where a quantitative strategy was evaluated (the greener a goal, the more satisfied it is). In URN, GRL is combined with Use Case Maps (UCM), a scenario notation that supports the modeling of causal sequences of responsibilities, optionally bound to an underlying structure of components, and their analysis through the simulation of scenarios. The modeling of stakeholder interests is necessary in SCPS and can be adequately covered with GRL.

The integration of goal and SysML modeling and analysis raises many challenges and research questions. Should goals be added to SysML, handled as a SysML profile, or handled externally? What relationships are worth defining and analyzing between goals and SysML model elements (e.g., the curvy arrows between the two models in Fig. 1)? Is there a specific role for requirements management systems (RMS) in that context? Should the integration depend on the tools available?

In this paper, we review important related work and tools related to these questions (section II). We then identify criteria for integrating goals with SysML (section III). We argue that SysML and URN are complementary and synergistic, and we briefly assess several possibilities for their integration (section IV), some of which we plan to investigate further in the future.

II. Related Work

A. Socio-Cyber-Physical Systems

Socio-cyber-physical systems stand at the intersection of cyber-physical systems and socio-technical systems (STS), often with “humans in the loop”. Whereas CPS tend to focus on technological aspects, SCPS are CPS that “pay especially attention to human stakeholders and their context-dependent behavioral aspect” [16]. Actors and goals are key concepts of STS and SCPS, and we anticipate their increased usage for emerging SCPS such as smart cities.

Goal models in such contexts can capture the (often conflicting) objectives of users and other stakeholders, but they can also express the objectives of organizations and other systems (especially in a SoS context), as well as context information. Zambonelli [39] also suggests the use of goals to model and reason about adaptive behavior, dynamic composition of services and systems, and collaborative services in SCPS.

B. URN, SysML, and Language Integrations

SysML and URN are two standards that have a rich history of integrations with other modeling languages.

The Use Case Map portion of URN was used by Miga et al. [23] to generate individual scenarios in the form of Message Sequence Charts (MSC), similar to UML 2 sequence diagrams. He et al. [17] also used UCM models to synthesize communicating state machine models in the System and Design Language (SDL), similar to UML 2 state machine diagrams. Blouin [10] explored the combined use of UCM and the Architecture Analysis & Design Language (AADL) [31].

The GRL part of URN was defined as a UML profile by Abid et al. [1]. Several integrations of GRL with languages other than UCM were proposed, first with the Business Process Model and Notation (BPMN) [26] by Losavio et al. [22] and Gröner et al. [15], and also with feature models (FM) by Alam et al. [4] and Asadi et al. [9]. Aspect-oriented extensions of URN (AoURN) were defined by Mussbacher et al. [25].

SysML, being a profile of UML, is often used in an integrated way with UML and/or with other profiles such as BPMN. In particular, Sena Marques et al. [33] used SysML with UML for performance requirements in embedded systems, whereas Briand et al. [11] used the same combination for traceability in a safety certification context.

The above integrations did not really support goal and systems modeling and analysis simultaneously. The idea of integrating goals to SysML was however explored by Ahmad et al. [2], where they used the KAOS goal-oriented language [35] and RELAX [38] to combine the elicitation power of GORE techniques with the semi-formal expressive power of textual, constrained requirements-specific languages. This first experience, applied to a supervision CPS, has been extended to general non-functional properties in [3]. The authors have made a first attempt to relate the GORE concepts of KAOS (e.g., contributions, impacts, and dependencies) to SysML concepts (e.g., Requirements, <<deriveRqt>>). While these experiments proved promising, they somewhat failed to provide an integrated method and set of concepts that would enable the extension of SysML with GORE capabilities in support of SCPS. In [37], Wanderley et al. have further attempted to improve human aspects by exploring the use of mind maps. Regarding methodology concerns, the approach proposed in [3] is limited by the fact that it integrates existing notations that are basically covering only one particular aspect of the requirements, i.e., with less coverage than what is offered by standard URN.

C. Tools

Given that SysML has been around for 15 years, numerous SysML tools are available (see http://sysml.tools/). Many are commercial, whereas a few are open source and free (e.g., Papyrus [13]). Some tools enable the analysis and simulation of SysML models, including coverage analysis of the requirements. Many tools support modeling based on multiple languages (e.g., SysML with UML and/or BPMN). To manage scalability, interoperability, and team work, some tool-supported SysML approaches [14][18] also allow for the synchronization of their requirements with an external requirements management system, e.g., IBM Rational DOORS [19]. Most SysML tools offer features for importing textual or tabular requirements (e.g., in Excel), and a few support standard requirements interchange formats such as ReqIF [27].

Of notable interest, the Analyst module of Modelio SA-SysML supports risks and goals in addition to requirements [24]. This tool's goal diagrams capture goals (as text), several dependencies (part of, refine, negative or positive influence, guarantee, measure, assignment), their properties, and to whom the goal is assigned (role or organizational unit). However, this solution is proprietary, and the analysis features are far more limited than what is available in GRL, especially in terms of quantitative analysis and support for indicators [7].

For URN, the free jUCMNav plugin for Eclipse is the best tool available at the moment [8], and also one of the best tools for goal modeling with *i**-like languages [5]. jUCMNav supports GRL strategies (qualitative, quantitative and hybrid [7]), indicators, contribution overrides, value ranges, strategy differences, various import/export mechanisms, and many other features beneficial to tradeoff and sensitivity analysis. jUCMNav also supports the UCM notation and links between UCM and GRL elements, including at the execution level. jUCMNav can also export MSC models from UCM execution traces.

For requirements management, jUCMNav can export URN models to IBM Rational DOORS [19] and keep models and textual requirements synchronized through DOORS reports and URN re-imports. In addition, users can select which model elements, associations, and attributes should be tracked in DOORS through a domain-specific language (MI-DSL [30]) used to specify such selection and generate the corresponding import library for DOORS. MI-DSL can actually be used for any language and tool, not just URN/jUCMNav.

III. Integration Assessment Criteria

The main objective of this research is the **improvement of requirements engineering activities with SysML through the integration of goal modeling**, with a particular focus on a SCPS context. We identified important criteria that should be used to assess the success of this integration:

- *C1.* Elicitation of, validation of, and general communication about goals and SysML (e.g., for SCPS).
- *C2.* Traceability/Completeness/Consistency analysis involving goals and SysML modeling elements.
- *C3.* Support for alternatives and tradeoff analysis.
- *C4.* Usability, e.g., through the minimization of the number of tools used and of necessary imports/exports.
- *C5.* Scalability, for handling large SCPS models.
- *C6.* Change management supporting SysML models, goal models, and textual requirements.
- *C7.* Concurrent modeling, to handle multiple modelers simultaneously (e.g., using a central repository to minimize duplication and inconsistencies).

C8. Ease of integration (e.g., via the use of existing tools).

There exist many tensions between these criteria (e.g., between C6 and C8), and we anticipate that only a subset of these criteria can be satisfied at once, at least in the short term. Note that cost (acquisition and maintenance) is a criterion that is orthogonal to the ones above and should be assessed separately. As such, it is not considered in this paper.

We also expect this list of criteria to be expanded in the future to cover new synergic types of analyses. For example: Can we support impact analysis for understanding the part of the SysML model needed to support a decision, without documenting all links to goals? Can we enable local decisions for a component of a SCPS without understanding the other partial impacts on higher-level goals? Can the integration of goals and SysML simplify documents for the certification of a SCPS?

IV. Integration Approaches

In an ideal world, goal-related concepts and diagrams would be integrated natively into a new version of the SysML standard, with full tool support for traceability and analysis. However, such vision would likely take many years and there are still too many open questions regarding how best to integrate these concepts that must first be answered. More pragmatic solutions satisfying the criteria from the previous section and enabling the exploration of goals/SysML integration are required in the meantime.

Different integration approaches can be investigated. In this paper, we are focusing on opportunities that take advantage of GRL/URN, as this is a standard language that includes strategies (for analysis) and indicators, with mature tool support (jUCMNav). The integration of GRL with UCM as part of URN, the existence of a UML profile for GRL, the availability of aspect-oriented extensions, and the integration of URN with feature models are further arguments that motivate this choice.

Figure 2 provides a technological context with which we can discuss categories of URN-SysML integration approaches. We foresee four potential combinations of high interest (coded *A1* to *A4*), based on whether goals should be handled internally or externally to the SysML tool, and whether an external RMS should be involved or not.

A. Goal Profile in SysML (A1)

Rather than modifying the SysML standard, a UML profile for GRL (e.g., Abid's [1]) could be added to a SysML tool that already supports UML (e.g., Modelio and Papyrus). In Fig. 2, everything would be done within the SysML tool. This would bring the benefit of having all modeling elements in the same environment, which would contribute positively to usability aspects. Change management would be offered by the tool, and requirements scalability would be that which is offered by the tool (which is often low in SysML tools). However, the graphical syntax of GRL might be difficult to support, the GRL analysis features would need to be developed within this environment, and UCM models would not be supported.

B. Goal Profile in SysML, with RMS (A2)

This approach is similar to *A1*, with similar challenges except that requirements scalability issues are handled through an external RMS such as DOORS. Many SysML tools already offer some integration with DOORS, either based on proprietary synchronization mechanisms or on the *Open Services Lifecycle Collaboration* (OSLC) standard, which supports loosely-coupled, point-to-point integration between tools [21]. OSLC also proposes a specific requirements management vocabulary that several modeling tools and requirements management systems already support. Whether these mechanisms are sufficient for goal models is a research question.

In Fig. 2, this approach involves the SysML tool and the RMS for external requirements, with integration based on link ②. As two separate tools are involved, usability will be slightly affected.

C. External Goal Support (A3)

In this approach, the goal model is managed externally in a separate tool (e.g., jUCMNav). In this option, GRL-level analysis can exploit all the strengths of jUCMNav, together with other features (integration with UCM, aspects, feature models, etc.). Although this topic is outside the scope of this paper, the presence of UCM also brings opportunities as a substitute for SysML use case diagrams and activity diagrams, and for generating sequence diagrams.

In Fig. 2, this approach involves the SysML tool and the URN tool, jUCMNav, with integration based on link ①. As two separate tools are involved, usability will be affected. This option would require much development effort as jUCMNav does not support OSLC and there is currently no import/export mechanism for traceability/synchronization between jUCMNav and SysML tools. Support for concurrent modeling and for scalability would be limited as well.

D. External Goal Support, with RMS (A4)

This approach uses all three tools from Fig. 2. The goal model is managed and analyzed by jUCMNav, the SysML model by a SysML tool, and textual requirements and traceability information (including that between goal model elements and SysML model elements) by an external RMS such as DOORS. This combination exploits the best the analysis features of each of these three tools. However, as three different tools are necessary, usability will be affected to a greater level.

jUCMNav already supports a default export mechanism to DOORS (link ③), and other subsets of URN can be defined through the use of MI-DSL [30]. In this context, generated DOORS queries and reports indicate potential impacts of changes to the SysML model or the textual requirements on the URN model. As mentioned in approach *A3*, OSLC is not currently supported by jUCMNav but this could be another (more effort-intensive) avenue to support this link.

As for approach *A2*, there are proprietary and OSLC-based solutions that exist between SysML tools and DOORS (link ②). As a third option, if no such mechanism exists, the MI-DSL-based export of SysML models to DOORS could also be considered, likely with a lower usability.

Although supporting links ② and ③ is sufficient, another variant of this approach could also support link ①, if OSLC is available in all three tools, for a higher usability level.

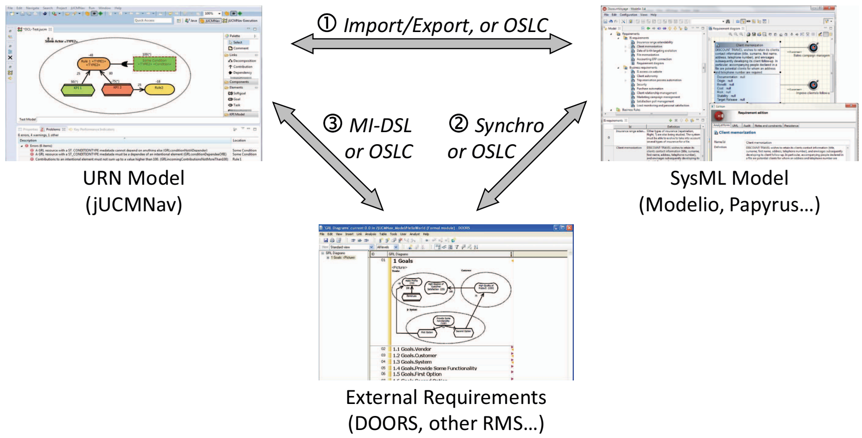

Fig. 2. Overview of relevant technologies and integration mechanisms

E. Preliminary Assessment of Criteria

In Table I, we provide a preliminary assessment of the four integration approaches against the criteria discussed in Section III (✓ for satisfied, × for unsatisfied, ± for partially satisfied, and ? for tool-specific). An evidence-based assessment of approaches is currently premature and remains part of our future work. Table I also gives an approximate assessment as different SysML tools have different strengths and weaknesses (e.g., in terms of usability or support for DOORS) that will impact concrete tool-specific assessments.

TABLE I. ASSESSMENT OF CRITERIA BY INTEGRATION APPROACHES

Approach	**Criteria (from Section III)**							
	C1	***C2***	***C3***	***C4***	***C5***	***C6***	***C7***	***C8***
A1	✓	✓	×	✓	±	?	?	±
A2	✓	✓	×	±	✓	✓	?	±
A3	✓	±	✓	±	±	×	×	×
A4	✓	✓	✓	×	✓	✓	±	±

At first glance, approach *A3* seems less globally interesting than the others. However, approach *A4* is globally promising, but the tradeoff is the low usability that would likely result from having to use three different tools.

V. CONCLUSIONS AND FUTURE WORK

This paper provided many arguments in favor of an integration of goal-oriented requirements engineering with SysML, especially in the context of emerging socio-cyber-physical systems. In particular, SysML can benefit from complementary modeling and analysis features found in GRL (and in URN in general). URN can obviously benefit from many missing views and concepts offered by SysML, leading to a mutual synergy. We also expect new beneficial types of analyses to emerge.

Yet, in practice, such integration is not obvious. This paper defined eight criteria that can be used to assess the success of a goal/SysML integration. A brief and preliminary assessment of four potential approaches against the criteria was performed and, if one wants to easily benefit from existing goal analysis features, then the approach *A4* might represent a good place where to start further investigation.

Many research questions were raised in this paper. In our future work, we plan to start evaluating several approaches starting with one based on *A4* that involves jUCMNav integrated with DOORS via MI-DSL, and a SysML tool that either already integrates with DOORS (e.g., Modelio) or that can easily be extended to do so via MI-DSL (e.g., Papyrus). We expect to collect empirical evidence about the usefulness of such integration through its use in a smart home/city case study. This will also give us a better understanding of how best to use GRL in a style compatible with SysML.

Later, we also plan on checking whether there is a SysML tool that can easily support a UML profile for GRL, not only at the modeling level but also at the analysis level (so we can re-implement some analysis features of jUCMNav in that SysML tool). This would allow us to investigate an approach like *A1* or *A2*. In parallel, we plan on investigating whether the additional use of UCM (and perhaps of AoURN and/or feature models) brings sufficient benefits to justify the integration of SysML with URN as a whole.

There are still many related challenges to be studied, which will hopefully be tackled by the requirements engineering research community.

ACKNOWLEDGMENT

DA thanks IRIT and the Université de Toulouse III for supporting him through their invited professor program. DA and LL are supported by Discovery grants of the Natural Science and Engineering Research Council of Canada. MB is supported by the King Abdullah Scholarship Program (KASP) and AAA by a scholarship from the Government of Libya.

REFERENCES

[1] M.R. Abid, D. Amyot, S.S. Somé, and G. Mussbacher, “A UML Profile for Goal-Oriented Modeling”, in SDL 2009: Design for Motes and Mobiles, LNCS 5719, Springer, 2009, pp. 133–148.

[2] M. Ahmad, J.-M. Bruel, R. Laleau, and C. Gnaho, “Using RELAX, SysML and KAOS for Ambient Systems Requirements Modeling”, in ANT/MobiWIS 2012, Procedia Computer Science 10, Elsevier 2012, pp. 474–481.

[3] M. Ahmad, N. Belloir, and J.-M. Bruel, “Modeling and verification of Functional and Non-Functional Requirements of ambient Self-Adaptive Systems”, Journal of Systems and Software, 107, 2015, pp. 50–70.

[4] O. Alam, J. Kienzle, and G. Mussbacher, “Concern-Oriented Software Design”, in MODELS 2013, LNCS 8107, Springer, 2013, pp. 604–621.

[5] C. Almeida, M. Goulão, and J. Araújo, “A Systematic Comparison of i* Modelling Tools Based on Syntactic and Well-formedness Rules”, in iStar 2013, CEUR Vol-978, 2013, pp. 43–48.

[6] D. Amyot and G. Mussbacher, “User Requirements Notation: The First Ten Years, The Next Ten Years”, Journal of Software, 6(5), 2011, pp. 747–768.

[7] D. Amyot, S. Ghanavati, J. Horkoff, G. Mussbacher, L. Peyton, and E.S.K. Yu, “Evaluating goal models within the goal-oriented requirement language”, Int. J. Intell. Syst. 25(8), 2010, pp. 841–877.

[8] D. Amyot, A. Shamsaei, J. Kealey, E. Tremblay, A. Miga, G. Mussbacher, M. Alhaj, R. Tawhid, E. Braun, and N. Cartwright, “Towards Advanced Goal Model Analysis with jUCMNav”, in ER Workshops 2012, Springer, 2012, pp. 201–210.

[9] M. Asadi, G. Gröner, B. Mohabbati, and D. Gašević, “Goal-oriented modeling and verification of feature-oriented product lines”, Soft. & Systems Modeling, 15(1), 2016, pp. 257–279.

[10] D. Blouin, Modeling Languages for Requirements Engineering and Quantitative Analysis of Embedded Systems, Ph.D. thesis, Université de Bretagne-Sud, France, December 2013.

[11] L.C. Briand, D. Falessi, S. Nejati, M. Sabetzadeh, and T. Yue, “Traceability and SysML design slices to support safety inspections: A controlled experiment”, ACM Trans. Softw. Eng. Methodol., 23(1), 2014, pp. 9:1–9:43.

[12] E. Cavalcante, T. Batista, N. Bencomo, and P. Sawyer, “Revisiting Goal-Oriented Models for Self-Aware Systems-of-Systems”, in 2015 IEEE International Conference on Autonomic Computing (ICAC), IEEE CS, 2015, pp. 231–234.

[13] Eclipse, Papyrus, Version 1.1.3, November 2015. Online: http://www.eclipse.org/papyrus/

[14] B. Friedland and G. Eschbacher, “Implementing SysML into a Feasible Systems Engineering Solution”, in Proc. Industrial and Systems Engineering Research Conference (ISERC), IIE, 2013.

[15] G. Gröner, M. Asadi, B. Mohabbati, D. Gašević, F.S. Parreiras, and M. Bošković, “Validation of User Intentions in Process Models”, in Advanced Information Systems Engineering, LNCS 7328, Springer, 2012, pp. 366–381.

[16] J. Hartmann, T. Makuschewitz, B. Scholz-Reiter, and E. Morosini Frazzon, “Towards Socio-Cyber-Physical Systems in Production Networks”, in 46th CIRP Conf. on Manufacturing Systems, Procedia CIRP 7, Elsevier, 2013, pp.49–54.

[17] Y. He, D. Amyot, and A.W. Williams, “Synthesizing SDL from Use Case Maps: An Experiment”, in 11th SDL Forum (SDL’03), LNCS 2708, Springer, 2003, pp. 117–136.

[18] E. Herzog, J. Hallonquist, and J. Naeser, “Systems Modeling with SysML – an experience report”, INCOSE International Symposium, 22(1), 2012, pp. 600–611.

[19] IBM, Rational DOORS, version 9.6, 2016. Online: http://www-03.ibm.com/software/products/en/ratidoor

[20] ITU-T, Recommendation Z.151 (10/12) User Requirements Notation (URN) - Language definition. Online: https://www.itu.int/rec/T-REC-Z.151/en

[21] D. Johnson and S. Speicher, Open Services for Lifecycle Collaboration, Core Specification Version 2.0, 2013. Online: http://open-services.net/bin/view/Main/OslcCoreSpecification

[22] F. Losavio, A. Matteo, and J.C. Guzmán, “Correspondencia Semántica entre los lenguajes BPMN y GRL”, Enl@ce: revista Venezolana de Información, Tecnología y Conocimiento, 8(1), 2011, pp. 11–30.

[23] A. Miga, D. Amyot, F. Bordeleau, D. Cameron, C.M. Woodside, “Deriving Message Sequence Charts from Use Case Maps Scenario Specifications”. SDL Forum 2001, LNCS 2078, Springer, 2001, pp. 268–287.

[24] ModelioSoft, Modelio Analyst, 2016. Online: https://www.modeliosoft.com/en/products/analyst.html

[25] G. Mussbacher, J. Araújo, A. Moreira, and D. Amyot, AoURN-based modeling and analysis of software product lines, Software Quality Journal 20(3-4), 2012, pp. 645–687.

[26] OMG, Business Process Model and Notation (BPMN), Version 2.0, formal/2011-01-03, 2011.

[27] OMG, Requirements Interchange Format (ReqIF), Version 1.1, formal/2013-10-01, 2013.

[28] OMG, Systems Modeling Language (SysML), Version 1.4, formal/2015-06-03, September 2015.

[29] OMG, Unified Modeling Language (UML), Version 2.5, formal/2015-03-01, 2015.

[30] A. Rahman and D. Amyot, “A DSL for importing models in a requirements management system”, in MoDRE 2014, IEEE CS, 2014, pp. 37–46.

[31] SAE International, Architecture Analysis & Design Language (AADL), standard AS5506B, 2012.

[32] G. Scanniello, M. Staron, H. Burden, and R. Heldal, “On the effect of using SysML requirement diagrams to comprehend requirements: results from two controlled experiments”, in EASE’14, ACM, 2014, Article 49.

[33] M.R. Sena Marques, E. Siegert, and L. Brisolara, “Integrating UML, MARTE and SysML to improve requirements specification and traceability in the embedded domain”, in 12th IEEE Int. Conf. on Industrial Informatics (INDIN), IEEE CS, 2014, pp. 176–181.

[34] A. Smirnov, A. Kashevnik, and A. Ponomarev, “Multi-level self-organization in cyber-physical-social systems: smart home cleaning scenario”, in 7th Industrial Product-Service Systems Conf. (PSS), Procedia CIRP 30, Elsevier, 2015, pp. 329–334.

[35] A. van Lamsweerde, Requirements Engineering: From System Goals to UML Models to Software Specifications, Wiley, 2009.

[36] E.S.K. Yu, D. Amyot, G. Mussbacher, X. Franch, and J. Castro, “Practical applications of *i** in industry: The state of the art”, in RE 2013, IEEE CS, 2013, pp. 366–367.

[37] F. Wanderley, N. Belloir, J.-M. Bruel, N. Hameurlain, J. Araújo, “Des buts à la modélisation système: une approche de modélisation des exigences centrée utilisateur”, in Actes du XXXIIème Congrès INFORSID, 2014, pp. 113–129.

[38] J. Whittle, P. Sawyer, N. Bencomo, B.H.C. Cheng, J.-M. Bruel, “RELAX: a language to address uncertainty in self-adaptive systems requirement”, Requirements Engineering, 15(2), 2010, pp. 177–196.

[39] F. Zambonelli, “Towards a General Software Engineering Methodology for the Internet of Things”, in arXiv, 21 Jan 2016, arXiv:1601.05569v1 [cs.SE].