User Requirements Notation: The First Ten Years, The Next Ten Years

(Invited Paper)

Daniel Amyot and Gunter Mussbacher
University of Ottawa, Canada
Email: {damyot, gunterm}@site.uottawa.ca

***Abstract*—The User Requirements Notation (URN), standardized by the International Telecommunication Union in 2008, is used to model and analyze requirements with goals and scenarios. This paper describes the first ten years of development of URN, and discusses ongoing efforts targeting the next ten years. We did a study inspired by the systematic literature review approach, querying five major search engines and using the existing URN Virtual Library. Based on the 281 scientific publications related to URN we collected and analyzed, we observe a shift from a more conventional use of URN for telecommunications and reactive systems to business process management and aspect-oriented modeling, with relevant extensions to the language being proposed. URN also benefits from a global and active research community, although industrial contributions are still sparse. URN is now a leading language for goal-driven and scenario-oriented modeling with a promising future for many application domains.**

***Index Terms*—Goals, Goal-oriented Requirement Language (GRL), modeling, review, scenarios, tools, Use Case Maps (UCM), User Requirements Notation (URN)**

I. Introduction

The *User Requirements Notation* (URN) is a modeling language that aims to support the elicitation, analysis, specification, and validation of requirements. URN is the first international standard to address explicitly, in a graphical way and in one unified language, goals and scenarios, and the links between them [106]. URN models can be used to specify and analyze various types of reactive systems as well as telecommunications standards and business processes. URN allows software and requirements engineers as well as business analysts to discover and specify requirements for a proposed system or process (or evolving ones), and analyze such requirements for correctness and completeness.

The kind of modeling supported by URN is different from the detailed specification of "how" functionalities are to be supported, as described with languages such as UML [146]. Here the modeler is primarily concerned with exposing "why" certain choices for behavior and/or structure were introduced, combined with an abstract view of "what" capabilities and architecture are required. The modeler is not yet interested in the operational details of internal component behavior or component interactions. Omitting these kinds of details during early development allows working at a higher level of abstraction when modeling a current or future software system, business process, or standard, and its embedding environment. Modeling and answering "why" questions leads us to consider the opportunities stakeholders seek out and vulnerabilities they try to avoid within their environment, whereas modeling and answering "what" questions helps identify capabilities, services, and architectures required to satisfy stakeholder goals.

Based on a systematic literature review, this paper provides a historical perspective on the development of URN together with trends related to future constructs and application domains for this notation. Such study is important at this point not only to appreciate the richness of URN and the substantial body of work that already exists, but also to step back, understand current trends, and anticipate future needs for evolving the notation in the right direction.

Section II introduces URN's basic concepts and notational elements, together with its standard analysis techniques. As it is important to understand why URN was created, a historical description of the origins of the notation is presented in Section III. Then, Section IV summarizes the main results of our literature survey, especially with regards to the sources of contributions to URN. In Section V, some of the main research contributions that have shaped URN in the past decade are categorized and reviewed, whereas section VI identifies current and future development activities and research areas related to URN for the next decade. Finally, section VII provides our conclusions.

II. Overview of URN

The User Requirements Notation standard combines two sub-languages [106]: the Goal-oriented Requirement Language for modeling actors and their intentions, and the Use Case Maps notation for describing scenarios and architectures. In this section, we give a brief overview of each of these sub-languages, supported by a simple URN model example that targets the evaluation of an architectural decision about where to put the data and the logic of the authorization service of a wireless system.

A. Goal-oriented Requirement Language (GRL)

GRL is a visual modeling notation for intentions, business goals, and non-functional requirements (NFR) of many stakeholders, for alternatives that have to be considered, for decisions that were made, and for rationales that helped make these decisions.

© 2011 ACADEMY PUBLISHER

A GRL goal graph is a connected graph of *intentional elements* that optionally reside within an *actor*. An *actor* (◌, e.g., Service Provider, Figure 1.a) represents a stakeholder of a system, or the system itself. A goal graph shows the non-functional requirements and business goals of interest to the system and its stakeholders, as well as the alternatives for achieving these high-level elements. Actors are holders of intentions; they are the active entities in the system or its environment who want goals to be achieved, tasks to be performed, resources to be available, and softgoals to be satisfied. *Softgoals* (⬭, e.g., Low Cost) differentiate themselves from *goals* (⬭, e.g., Determine Data Location) in that there is no clear, objective measure of satisfaction for a softgoal whereas a goal is quantifiable, often in a binary way. Softgoals are often more related to NFR, whereas goals are more related to functional requirements. *Tasks* (⬡, e.g., Install Service Node) represent solutions to (or *operationalizations* of) goals or softgoals. In order to be achieved or completed, softgoals, goals, and tasks may require *resources* (□, e.g., Service Node) to be available.

Various kinds of *links* connect the elements in a goal graph. Decomposition links allow an element to be decomposed into sub-elements (┼—, e.g., High Performance is decomposed into Maximum Hardware Utilisation and High Throughput). AND, IOR, as well as XOR decompositions are supported. Contribution links indicate desired impacts of one element on another element (→, e.g., Minimum Changes to Infrastructure contributes to Low Cost). A contribution link has a qualitative contribution type (Figure 1.b) or a quantitative contribution (an integer value between -100 and 100). Correlation links (⇢) are similar in nature, but describe side effects rather than desired impacts. Dependency links model relationships between actors (—◗—, e.g., System depends on Vendor for Service Node).

GRL supports reasoning about goals and requirements, especially NFR and quality attributes, as it shows the impact of often conflicting goals and various global alternative solutions proposed to achieve the goals. A GRL *strategy* describes a particular configuration of alternatives in the GRL model by assigning an initial qualitative satisfaction level (Figure 1.c) or a quantitative one (an integer value between -100 and 100) to some of the intentional elements in the model (indicated by a star (*) and a dashed outline), often leaves in the GRL graph. An *evaluation mechanism* propagates these low-level decisions regarding alternatives to satisfaction ratings of high-level stakeholder goals and NFR. Strategies can therefore be compared with each other to help reach the most appropriate trade-offs among often conflicting goals of stakeholders. A good strategy offers rationale and documentation for decisions leading to requirements, thus providing better context for systems and software engineers while avoiding unnecessary re-evaluations of worse alternative strategies. Color coding of the intentional elements also reflect their satisfaction level (the greener, the more satisfied).

GRL takes into account that not all high-level goals and NFR are equally important to a stakeholder. Therefore, GRL supports the definition of an *importance* attribute for intentional elements inside actors (again quantitative or qualitative, and shown between parentheses, e.g., 50 for Low Cost). This attribute is also taken into account when evaluating strategies for the goal model, resulting in satisfaction levels measured at the actor level (e.g., 32 for the Service Provider).

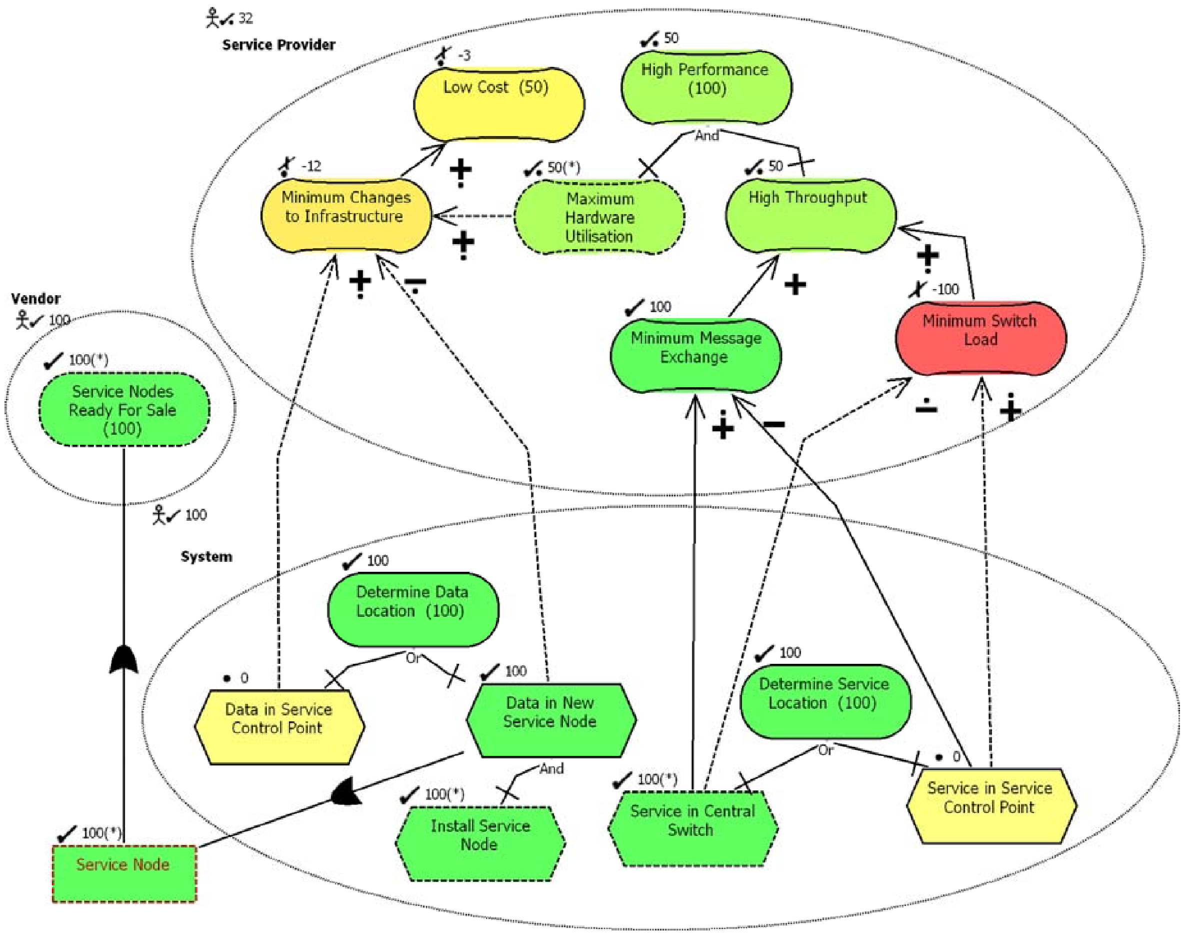
The current URN standard does not enforce a specific evaluation mechanism as GRL can be used in different ways by different modelers, e.g., for qualitative evaluations or quantitative ones, but provides three non-normative examples of evaluation algorithms. A hybrid algorithm combining qualitative contributions and quantitative satisfaction levels is used for one strategy in Figure 1.a. A different strategy would lead to different results, enabling comparisons and documenting decisions.

B. Use Case Maps (UCM)

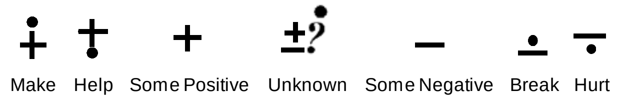
The UCM visual scenario notation focuses on the causal flow of behavior optionally superimposed on a structure of components. UCM depict the causal interaction of architectural entities while abstracting from message and data details.

The basic elements of the UCM notation are shown in Figure 2. A *map* contains any number of paths and components. *Paths* express causal sequences and may contain several types of path nodes. Paths start at *start points* (●, e.g., StartConnection) and end at *end points* (▌, e.g., Done), which capture triggering and resulting conditions respectively. *Responsibilities* (✕, e.g., LogReject) describe required actions or steps to fulfill a scenario. *OR-forks* (—<), possibly including guarding conditions such as [NotOk], and *OR-joins* (>—) are used to show alternatives, while *AND-forks* (—⊏) and *AND-joins* (⊐—) depict concurrency. Loops can be modeled implicitly with OR-joins and OR-forks. As the UCM notation does not impose any nesting constraints, joins and forks may be freely combined and a fork does not need to be followed by a join. *Waiting places* (●) and *timers* (◷) denote locations on the path where the scenario stops until a condition is satisfied.

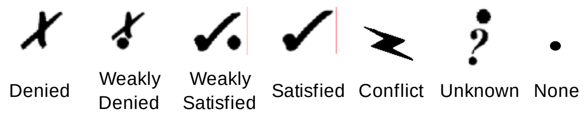
UCM models can be decomposed using *stubs* that contain sub-maps called *plug-in maps* (see Figure 2.b and c). Plug-in maps are reusable units of behavior and structure. *Plug-in bindings* define the continuation of a path on a plug-in map by connecting in-paths and out-paths of a stub (*IN1* and *OUT1* in Figure 2) with start and end points of its plug-in maps, respectively. Plug-in bindings also describe the relationship of components on the parent map with the ones on the plug-in map (e.g., the parent component of the plug-in map in Figure 2.c refers to a component in the parent map, ControlFunction in this example). A stub may be *static* (◇), which means that it can have at most one plug-in map, whereas a *dynamic* stub (◇, e.g., Authorization) may have many plug-in maps that can be selected at runtime according to a *selection policy*. In Figure 2, the two plug-in maps represent alternative ways of supporting authorization, with different locations for the data and the logic of the service (i.e., different allocations of responsibilities to components).



(a) GRL graph for a system with two stakeholders

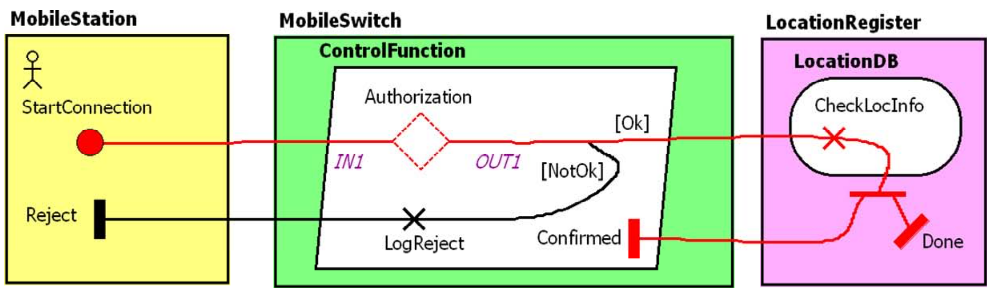


(b) GRL Contributions Types

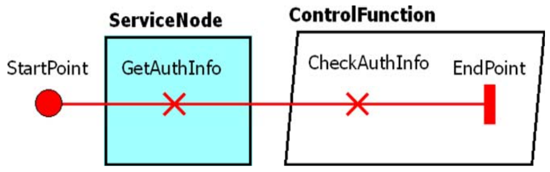


(c) GRL Satisfaction Levels

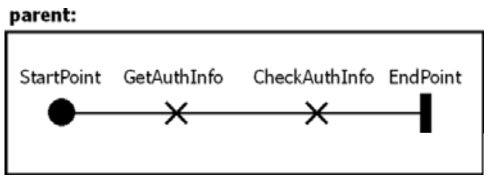
Figure 1 GRL example: Where should the data and the service be located in the system?



a) Top-level map: Connection request to a mobile switch



b) Plugin 1: Service in mobile switch, data in external service node



c) Plugin 2: Service and data in mobile switch

Figure 2 UCM example: Connection scenario (a), with two potential architectural solutions (b and c) for the authorization service.

Components (□, e.g., MobileSwitch) are used to specify the structural aspects of a system. Map elements which reside inside a component are said to be bound to it. Components may contain sub-components and have various types and characteristics. For example, a component of kind *object* (⬭, e.g., LocationDB) does not have its own thread of control whereas a component of kind *process* (▱, e.g., ControlFunction) does. A component of kind *actor* (웃, e.g., MobileStation) represents someone or something interacting with the system under design.

UCM support the definition of *scenarios* including pre- and postconditions. A scenario describes a specific path through the UCM model where only one alternative at any choice point is taken. The UCM notation supports a simple but formal data model that can be used to formalize the conditions at selection points (e.g., dynamic stubs and OR-forks). Responsibilities can also include code that modifies the values of the variables used in this data model. A scenario definition can hence be expressed with initial values for these variables, combined with a sequence of start points being triggered.

Given the definition of a scenario or combination of scenarios, a *path traversal mechanism* can highlight the scenario path being simulated. Figure 2 shows in red the paths traversed for the scenario where the service logic remains in the mobile switch but the service data is located in a new external service node (which corresponds to the strategy being evaluated for the GRL model in Figure 1.a), and where the authorization is OK. The traversal mechanism essentially provides the operational semantics of the UCM language. It also turns the scenario definitions into a test suite for the UCM model, which is especially useful for regression testing as the model evolves.

Different elements in a UCM model can also be annotated with specific performance information, enabling early performance analysis at the requirements level. For example, resources can be defined and components assigned to them, selection points can include probabilities, responsibilities can specify demands on resources, and start points can include workload definitions.

The UCM notation enables a seamless transition from the informal to the formal by bridging the modeling gap between goal models and natural language requirements (e.g., use cases) and design artefacts, in an explicit and visual way.

C. Integration of Goals and Scenarios in URN

Modeling goals and scenarios is complementary and may aid in identifying additional or spurious goals and scenarios, thus contributing to the completeness and accuracy of requirements. In the language, *URN links* (▶) can connect any two URN model elements, establishing traceability links that further tighten the relationship between GRL and UCM models while enabling completeness and consistency analysis.

The URN language also supports the concept of *metadata* in the form of name/value pairs that can be associated with any URN model element. This allows for domain-specific extensions to be added to URN and exploited by specialized tool support.

III. Pre-URN History (1990-1999)

The roots of URN go back to the early 90's. Use Case Maps originate from Carleton University, where Buhr used them as a high-level notation in their project Design of Object-Oriented Real-time Systems (DOORS — http://www.sce.carleton.ca/rads/doors.html). Vigder's early work on *design slices* [186] used a scenario-like notation with a connection to the LOTOS formal specification language [99]. Buhr then coined the term *timethread* as a name for this graphical notation, which was used in a few papers and theses until the release, in 1995, of a seminal book co-authored with Casselman where the term *Use Case Maps* emerged [52]. This book focused on the application of UCM to object-oriented systems, with an emphasis on role modeling concepts developed in Casselman's thesis [56]. Another important milestone for the UCM notation was the publication of a revised and more powerful version of the language in a major journal [49]. In those years, typical applications that were explored with this notation included design activities [13][40] (including architecture [46][49] and patterns [47][50]), performance analysis [170], and the modeling of telecommunication [8][10], agent-oriented [51][68], and e-commerce [79] systems. Miga provided tool support for the creation and analysis of UCM models [129], based on an earlier prototype from Carrière (UCMEdit, discussed in [113]). This multi-platform UCM Navigator tool (UCMNAV) was used in academia and industry mainly between 1998 and 2005 [181].

Work on goal modeling for requirements, agents, and organizations that was being done at that time at the University of Toronto guided the development of the GRL language. The syntax of GRL is in fact based on the *i** framework described in Yu's thesis [197], which was developed for describing strategic relationships in organization models. The reasoning mechanisms behind another goal-oriented notation, namely the *Non-Functional Requirements (NFR) Framework* (best described in the seminal book of Chung, Mylopoulos, Nixon, and Yu [62]), also inspired the evaluation and propagation mechanisms now found in GRL. Tool support for modeling and analyzing goal models (in *i**, the NFR Framework, and GRL) was then provided by the Java-based *OME 3*, Yu's Organization Modelling Environment [198].

The idea of creating a new standard notation was first proposed in 1999 by Visser and Hodges from Nortel, as they were deeply involved in standardization activities with the International Telecommunication Union (ITU-T) and with the Wireless Intelligent Network initiative [96]. Through collaborative research projects with Logrippo (University of Ottawa) and his team [10][21], it was observed that UCMs would likely be more appropriate than natural language and Message Sequence Charts (MSC) [104] for early descriptions of wireless telecommunication features. Monkewich (also from

Nortel) brought the idea of creating a Use Case Maps standard to the language experts at ITU-T, who then suggested renaming it to “User Requirements Notation”. This potential standard captured the attention of another Ottawa-based company, Mitel, where Pinard, Weiss, Gray, and Mankovski had also used UCM for modeling telecommunications features. However, they were also interested in *i** and the NFR Framework for goal-oriented and agent-based modeling. They were collaborating with the University of Toronto on projects that led to the creation of a new goal modeling language by Yu and Liu, which became the first version of GRL [126].

Gray and the Mitel experts expected great benefits in combining goals with scenarios and regarded this combination essential for the understanding of highly dynamic and reflective systems, and for feature personalization. This potential integration led to the introduction of dynamic stubs in the UCM notation in the mid-90’s. Gray convinced the Nortel experts and other stakeholders to revise the URN proposal as a Canadian contribution to ITU-T that would include both GRL and UCM. This was then accepted as a new work item at ITU-T in 2000, and Hodges became the first Rapporteur for the URN question.

IV. Systematic Literature Review

A. Methodology

Inspired by the work of Kitchenham *et al.* [43][118], we did a systematic literature review targeting the following three questions:

- Who contributed to the development of URN?
- What research contributed to the development of URN?
- What are the current and future development activities and research areas related to URN?

In July 2010, we used five major search engines for publications in computer science and engineering (IEEE Xplore, ACM Digital Library, Google Scholar, SpringerLink, and Scopus). Our query was simply `"User Requirements Notation" OR "Use Case Map" OR "Goal-oriented Requirement Language"`, which covered the essential keywords. The URN, GRL, and UCM acronyms were not included because an early assessment led us to believe they were polluting the results without really identifying more valid citations. Over 700 references were collected in the end, mostly coming from Google Scholar. These were combined with the references already present in the URN Virtual Library [182].

We restricted the results to scientific publications appearing in journals, conferences, workshops, books, and theses. Furthermore, we excluded papers that:

- Only cited URN (or GRL/UCM) to acknowledge its existence or to discuss it in a comparison.
- Simply used URN (or GRL/UCM) to illustrate some requirements or design (e.g., with a few diagrams), without discussing the usage of the language itself.
- Focused on the “other” Use Case Map concept developed by Constantine and Lockwood [63], which is a variant of UML use case diagrams used to model the interrelationships among use cases (different from URN’s Use Case Maps).

We finally included seminal work produced prior to the use of the terms UCM [56][186] and GRL [62][197].

B. Contributors and Contributions

Our selection led to a total of 281 scientific publications related to research on and with URN. More specifically, we have found 38 journals papers, 183 conference and workshop papers, 15 books and book chapters, as well as 45 theses (13 Ph.D., 31 Master’s, and 1 B.Sc.) The URN Virtual Library was updated to include the 31 publications that were missing prior to this literature review. Figure 3 shows the distribution of our four types of publications over the years.

To answer our first question, this data shows that there were 263 different authors involved (with an average of 2.7 authors per paper, often from different locations). Given the origins of URN, it is not surprising to see that the majority of the papers (66%) and theses (80%) published since 1992 include co-authors from Canada, especially from the University of Ottawa and from Carleton University (see Table I). Actually, all papers and theses prior to 1999 came from Canada.

TABLE I.
Number of Co-Authors per Country

Country	Papers	Theses	Total
Canada	162	36	198
U. of Ottawa	103	21	124
Carleton U.	64	10	74
Concordia U.	15	4	19
U. of Toronto	7	1	8
Other places	19		19
The Netherlands	10	4	14
UK	11	1	12
USA	9		9
Japan	8		8
Hungary	6		6
Norway	4	1	5
Italy	4		4
Australia	4		4
Brazil	4		4
Germany	3	1	4
Argentina	3	1	4
Spain	3		3
China	3		3
Belgium	3		3
Portugal	3		3
Korea	2		2
Viet-Nam	2		2
South Africa	2		2
United Arab Emirates	2		2
Switzerland	1	1	2
Serbia	1		1
Latvia	1		1
Poland	1		1
Venezuela	1		1
Libya	1		1
Thailand	1		1
Sweden	1		1
Chile	1		1

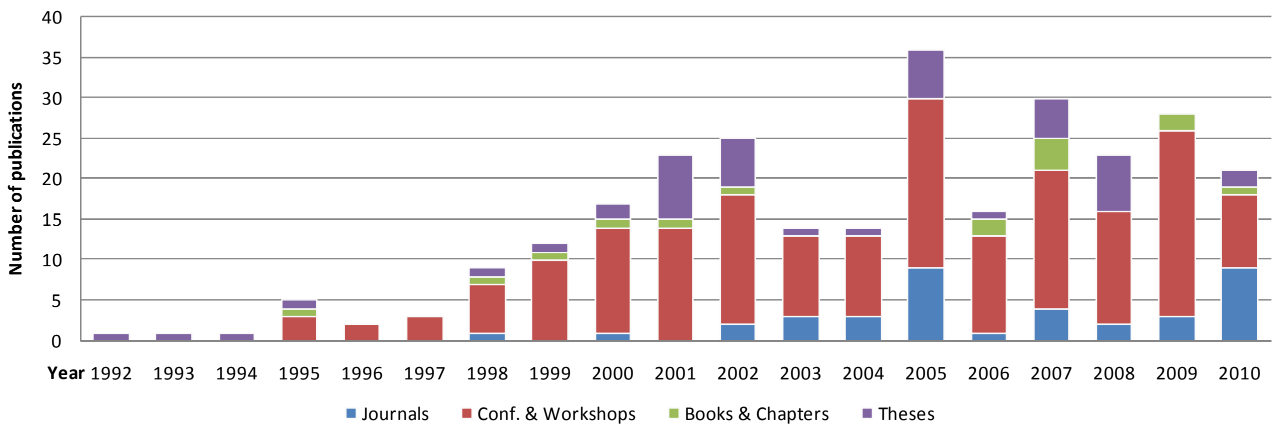


Figure 3 Number of URN-related scientific publications per year.

However, work on URN then started to draw international involvement. Between 1999 and July 2010, we observed that 43% of the papers had *at least one* co-author from outside Canada, 34% of the papers had *all* their co-authors from outside Canada, and 23% of the theses were from outside Canada. Collaboration between Canadian and non-Canadian authors has also increased substantially over the past four years. As shown in Table I, researchers and industrial participants from over 29 different countries on all continents have contributed publications on URN. Incidently, in our data set, we also detected papers written in seven languages other than English: Chinese (4), French (3), Spanish (3), Serbian (2), Japanese (2), Korean (1), and German (1).

Although our results support that scientific contributions related to URN are numerous and international, industrial contributions are still sparse: only 22 papers (9% of our data set) involved co-authors with industrial affiliations, mainly from the telecom industry (Nortel, Mitel, Cisco, and others). This may partially be explained by the fact that we excluded many papers where URN was simply used.

C. Bias

The content of the URN Virtual Library and our selection of papers are somewhat biased towards UCM because the work related on UCM is all included in URN's, whereas prior and subsequent work on the NFR Framework and especially on *i**, on which GRL is initially based, is not included. *i** is different from GRL and has a community of its own (e.g., four *i** workshops and 55 research teams are listed so far on the *i** Wiki [97]). This is also reflected in the numbers of references we collected (e.g., from Google Scholar, we found 442 for UCM, 294 for URN, and 186 for GRL). Consequently, many of the topics discussed in the next two sections focus on UCM and fewer will address GRL exclusively.

Note that we have done the collection and filtering of the papers ourselves. However, to mitigate internal bias, an exhaustive search using Google Scholar and other engines (see previous section) was performed. Given that we have been involved with URN since its inception, that we co-authored about a third of the scientific papers on URN, that we co-edited the standard and that we are responsible for its evolution at ITU-T, we believe we are uniquely positioned to perform a rigorous assessment of the past and future research on URN.

V. The First Ten Years (2000-2009)

After ITU-T's approval of a new question on a User Requirements Notation, the URN standardization took another eight years. Cameron (Nortel) took over from Hodges as Rapporteur in 2001, followed by Amyot (University of Ottawa) in 2002. The first standard (Z.150, in 2003) described the goals and requirements for the URN language [105]. In 2008, the definition of the URN language itself (including a URN metamodel based on the new meta-metamodel of ITU-T Z.111 [103]) finally became available [106].

This section summarizes some of the most important research contributions that have led to this standard, together with emerging application domains.

A. Specification and Validation of Protocols and Services

One of the main drivers behind the creation of URN was to enable standards bodies such as ITU-T to specify and perform early validation of new telecommunication protocols and services. UCM models, in particular, provide a view that abstracts from messages and potentially from component architectures, which simplifies the description of services. This view also helps multiple stakeholders such as vendors and carriers (with conflicting agendas and investments in different legacy networks) reach consensus on the essence of these services.

There were indeed several services specified with UCM and proposed for standardization, in the early 2000s, to ITU-T, the Telecommunications Industry Association (TIA), and the Internet Engineering Task Force (IETF). In the literature, we notice the International Mobile Telecommunications-2000 (IMT-2000) [27] and other mobile wireless protocols [25][26], the Open Shortest Path First (OSPF) routing protocol [132], Mobile IPv6 [187], Call Name Presentation (CNAP) [196], and GPRS mobile group call [20], among others. All of these specifications helped further shape the notation as well as stylistic guidelines for its application to telecom services.

More recently, URN was revisited in a service engineering context. Amyot *et al.* [12] described a URN-based approach for specifying Next Generation Network (NGN) services, where service models (combining GRL, UCM, and UML views) provide information and mechanisms that help dynamic composition and adaptation at runtime. In his thesis, Castejón focuses more on the concept of collaboration for compositional service descriptions, with both UML and UCM [58]. These two approaches to service engineering helped explore and understand the synergy between URN and UML collaborations.

B. Multi-Agent Systems

In the past decade, there was quite a bit of attention devoted to the use of URN in the domain of agent systems, beyond what was already done in the 90's by Buhr and others as part of their *High-Level Design and Prototyping of Agent Systems* project [51][68].

Bush *et al.* [53] introduced their Styx agent methodology, where UCM are used to capture high-level system processes. A similar use of UCM is discussed in the approaches proposed by Araya and Antillanca [28] and by Abdelaziz *et al.* [2], the latter with an interesting application to a medical diagnosis system. In his thesis [1], Abdelaziz further enhanced his Multi-Agent System Development approach by integrating UML use case, activity, and sequence diagrams (and to some extent GRL dependency models) with the UCM view. Lavendelis and Grunspenkis proposed the MASITS tool, similar in intent to Buhr's [51], for capturing several views of agent systems, including goal and use case views à la URN. Saleh and Al-Zarouni [165] described the use of GRL for capturing non-functional requirements for mobile agent systems. Amyot *et al.* [14] specified and analyzed, with UCM, an agent-based telecommunication system being built by an industrial partner.

While Billard captured a collection of eight agent interaction patterns with UCM, with an analysis of their performance [34], Weiss exploited the NFR Framework (at the basis of GRL) to describe and exploit the relationships between patterns used in the design of agent systems [188].

C. Web Applications and Web Services

In the mid-2000s, URN was also used in the context of Web applications. Yu and Liu, following seminal work where they first proposed an iterative methodology that combines GRL and UCM and demonstrate the complementary nature of these two views [124], successfully specified a Web-based training application with URN [125]. Around the same period, Kaewkasi and Rivepiboon also proposed a methodology for Web application modeling, but this time based on a combination of UCM and UML [111]. Around 2005, Weiss started to describe patterns (partly with UCM) for Web applications [189], while also exploring with others the UCM-driven testing of Web applications [22].

Web services also captured the attention of URN contributors. Weiss and Esfandiari provided preliminary results on the analysis of personalized Web services where service goals are specified with GRL and service functionalities with UCM [191]. van der Raadt explored Web services from a business perspective with the Business-oriented Approach Supporting web Service Idea Exploration (BASSIE) methodology. BASSIE combines three types of models: a) *i** (instead of GRL) for describing strategic goals and the impact of service realization alternatives, Gordijn's *e*^{3}*value* framework [77] (originating from UCM) for evaluating alternatives based on their profitability, and UCM for describing other details of the services.

These two approaches to the development of Web applications and services helped clarify the relationships that exist between GRL and UCM views, with an impact on the inclusion of the concept of URN link in the standard.

D. Formalization

The URN standard describes the URN abstract and concrete syntaxes formally, together with well-formedness constraints. However, the semantics is currently described more informally using traversal requirements for UCM (with which many algorithms could comply) and with propagation requirements for GRL (with, again, many potential evaluation algorithms). Hence, these textual requirements do not fully alleviate the risk that different tools could implement different semantics while still satisfying the standard's requirements.

It was judged premature to agree on a unique semantics to the notation in the standard, although many had already been proposed, especially for UCM. One of the first attempts was done by Amyot and Logrippo [8], and was based on the LOTOS process algebra [99]. The mapping from UCM to LOTOS was further explored by Guan [82], who also provided a compiler for UCMNav models. This mapping was used in 9 theses and 13 publications, and contributed to the understanding of UCM behavior.

van der Poll *et al.* proposed an initial, informal mapping from UCM to Z in order to formally analyze models capturing user interface scenarios [183]. This work was further extended by Dongmo [67], who defined a framework to derive Object-Z [176] class schemas from UCM models. Similarly, Truong *et al.* [180] proposed a mapping from UCM to the B formal method [171] in order to support the verification of component behavior against the UCM scenario requirements. To facilitate the analysis of component interfaces and composition, de Bruin explored a mapping from UCM models enhanced with interface information to the object-oriented programming language BCOOPL [64].

Hassine *et al.* also provided a formal semantics for UCM, but this time based on Abstract State Machines (ASM, [38]), with tool support for simulation [88]. They further investigated the use of quantitative time constraints in UCM models with a UCM extension called *Timed Use Case Maps*, for which they provided additional semantics based on more appropriate formalisms, namely timed automata [90], clocked transition systems [89], and again ASM [86]. Hassine's thesis [85] is the best document where these extensions and semantics are used, and a recent survey provides a comparison with related timed scenario languages [92].

There is no formal semantics for GRL at this point. However, the initial description of GRL [126], which did not include a meta-model at the time, was evaluated by Heymans *et al.* [95] from an ontological perspective. Some of their conclusions were actually taken into consideration in the definition of the URN metamodel. Ayala *et al.* also provided an interesting analysis of GRL compared to other goal-oriented modeling languages, but again this is based on the original version proposed in 2001 (not the standard definition). A comparison between *i** and van Lamsweerde's KAOS is offered by Matulevičius *et al.* [127] and also by [17] from an analysis point of view. The *i** Wiki [97] is also an interesting source of information on formalization for related goal-oriented languages. For comparisons between UCM and other scenario languages (too numerous to mention them all here), the reader is referred to the studies of Saiedian *et al.* [164], Amyot and Eberlein [16], and Mussbacher and Amyot [136].

E. Transformations to Design Models

Scenario models such as those specified with UCM represent a good basis for transformations to more detailed design representations. Such transformations enable the generation of design artifacts with less effort and, yet, higher consistency with the requirements.

Bordeleau, Buhr, and Cameron were among the first to explore systematic relationships and transformation between UCM models and (High-level) Message Sequence Charts [41]. Miga *et al.* [130] have then demonstrated that lengthy scenarios resulting from UCM path traversals can be transformed to MSC in order to visualize them in a more scalable and linear form. They implemented this transformation in the UCMNAV tool [181]. The main challenges in this transformation is to infer or synthesize necessary messages ensuring that causal relationships between responsibilities in different components are correctly supported, and to handle the well-formedness rules of a linear scenario representation like MSC, which are stricter than the general graph representation of UCM. New results partly addressing these challenges were provided by Amyot *et al.* [15]. However, the best implementation so far is the one now found in the jUCMNav tool [110], as provided by Kealey in his thesis [114]. Along the way, Kealey redefined and greatly improved the power, flexibility, and robustness of the path traversal algorithm initially proposed by Miga, and this contribution had a major impact on the definition of the path traversal rules now found in the standard [114].

The synthesis of state machines from scenarios was a topic of high interest in the 2000's [16][164]. Bordeleau, Corriveau, and Selic were among the first to provide guidelines for the transformation of UCM models to hierarchical state machines [42]. Sales and Probert also proposed transformation guidelines [165], only this time the target language was SDL [102]. He *et al.* [93] explored an automatic transformation from UCM to SDL via the intermediate generation of MSC from UCM and the synthesis of SDL models from these MSC (based on a commercial MSC-to-SDL synthesizer). Castajón also reported on an experiment on the synthesis of state machine behavior from UML collaborations whose dependencies are captured with UCM models [57].

On the goal side, we notice the combination of GRL and a security extension to UML (UMLsec) proposed by Saleh and Elshahry to model security requirements across goal and design views [166]. Abid *et al.* also proposed a UML profile for GRL, hence enabling the integration of a GRL view in UML design models [3].

F. Feature Interaction Analysis

The various formalisms used to analyze URN models, as seen by the many transformations and formalizations discussed in the previous sections, are important to support the detection of undesirable interactions between features or service descriptions, a problem well known in telecommunications and other domains [55].

Amyot *et al.* used a mapping from UCM to LOTOS, combined with a testing approach, to support the rigorous detection of interactions between telecommunication features [8][14]. Due to the numerous test cases that have to be checked for large sets of features, the need for identifying situations where interaction tests are needed became apparent. Nakamura *et al.* hence proposed an interaction filtering approach based on the stub/plug-in structure of UCM models and formalized with stub configuration matrices [144]. This technique helped reduce the number of test cases needed to detect undesirable interactions by focusing on interaction-prone combinations. This seminal work led to various improvements by Cheng *et al.* [61] and Zhang and Liu [200] in terms of the required pre-conditions, and by Leelaprute *et al.* [121] who added a second phase for the generation of error-prone scenarios from the interaction-prone configurations. Hassine also adapted this filtering technique to identify interaction-prone combinations targeting LOTOS specifications, which were then checked formally using tests and goal-oriented executions [84]. In his thesis, Gorse proposed a different filtering technique, this time based on a logic representation of the feature requirements in Prolog. The filtering results are used for testing a LOTOS specification that formalizes features modeled with UCM [80].

Shiri *et al.* [174] combined UCM with Birkoff's Formal Concept Analysis [35] to assist maintainers in identifying feature modification impacts at the requirements level, and minimizing the need for regression testing.

Weiss and Esfandiari studied the feature interaction problem in terms of functional and non-functional interactions [192]. They used GRL to analyze conflicting goals, tradeoffs between softgoals, inadequate interfaces, ownership and policy issues, and resource contention. They also used UCM to analyze concurrency issues, violations of assumptions, and incorrect invocation ordering. This work led to the first classification of undesirable interactions for Web services.

More recently, Mussbacher *et al.* [143] studied semantic-based interactions in aspect-oriented models. Their approach differs from the syntactic approaches like filtering, and is more lightweight than detection methods that rely on the use of underlying formal languages. This approach requires the manual annotation of aspects with domain-specific markers, and a GRL model that specifies how markers from different domains influence each other. Automated analysis can then be used both to highlight semantic aspect conflicts and to trade-off aspects. This approach is demonstrated on academic and industrial examples that use aspect-oriented extensions of UCM [134] and other languages.

G. Performance Analysis

In URN, modelers may supplement UCM elements with standard performance annotations to describe resources associated with components, demands of responsibilities, workloads on scenario start points, allocations of UCM components to devices, and probabilistic behavior at selection points [106]. These annotations are not taken into consideration for the path traversal mechanism, but they can be used in transformations of UCM models to specialized performance models. This enables performance analysis from URN requirements models, before serious barriers to performance are frozen into the design and implementation.

This part of the standard was strongly influenced by Woodside and his research team at Carleton University. In his PERFECT method, Scratchley used annotated UCM for evaluating concurrency architectures for a system that executes a given set of scenarios [169]. His annotations included timestamps and response-time requirements (implemented in the original UCMNAV tool), which were replaced in the URN standard by more generic metadata and URN links.

The generation of Layered Queueing Network (LQN) performance models [177] directly from UCM models was first explored and prototyped by Petriu [152][153]. LQN performance models can be used as a basis for exploring the performance solution space of a system. Different kinds of analyses (e.g., sensitivity, scalability, concurrency, and configuration) can be performed through the use of LQN solver and simulation tools. Siddiqui *et al.* [175] improved upon this approach to consider the notions of budget and completions in the analysis of performance, while Liu focused on a multi-level methodology, with application to large presence systems [123]. Wu and Woodside explored the hybrid use of LQN and generalized stochastic Petri Nets for the performance analysis of annotated UCM models [194]. A good summary of the UCM-LQN performance engineering vision is found in [154].

The original annotations influenced the early development of the UML profile for schedulability, performance, and time [147]. More recent work on the development of the Core Scenario Model (CSM) representation [155], led to new annotations that are now part of the URN standard. These annotations have also evolved in synergy with the creation of the new UML profile for real time and embedded systems (MARTE) [148].

CSM's purpose is to capture the essence of a range of scenario notations (e.g., from URN and UML) and enable simple transformations to various target formalisms (e.g., LQN, regular queueing networks, and stochastic Petri Nets), hence reducing the number and complexity of tools needed to analyze various aspects of the same system. Accordingly, newer URN-based approaches now target the generation of CSM models rather than LQN directly. A transformation from UCM to CSM was defined by Zeng in his thesis [199], with an implementation in UCMNAV. This transformation was adapted by Sincennes and others and is implemented in jUCMNav. One of the main benefits of this approach is that the acquisition and release of resources is inferred implicitly from UCM models rather than requiring them to be defined explicitly as in profiled UML models. This simplifies substantially the creation and maintenance of models. Transformations from CSM models to LQN models and other types of performance models are discussed in [155] and are now supported by prototype tools.

Other types of software performance analysis based on UCM do not make use of the standard performance annotations. Billard used his own queueing simulator to analyze the UCM model of an object-oriented operating system [33], whereas Hassine used Timed Use Case Maps and a mapping to ASM to analyze resource allocation, worst-case time execution, and schedulability issues in an automatic protection switching feature [87].

Cai and Yu [54], on the other hand, investigated a GRL-based approach for qualitatively addressing and refining performance requirements. Operationalizations of such requirements are linked to UCM scenarios.

H. Architecture Evaluation

By combining goals with scenarios, URN provides a unique perspective on the evaluation of architectures. The previous section discussed performance-oriented approaches that often require the generation of mathematical performance models, where the quantitative parameters are difficult to choose and set. de Bruin and van Vliet explored a more qualitative approach to architecture evaluation, where a feature model describes the alternatives and refinements of the problem domain, whereas the solution domain is captured with a UCM model (with stubs and plug-ins) [65]. Links between the two models enables the evaluation and selection of an appropriate architecture, with its behavioral description. This approach shares similarities with the method of Liu and Yu [124], where a GRL goal model is used to capture actor intentions and coarse-grain alternatives, whose operationalizations are linked to the UCM view.

Many surveyed approaches also focus on specific architectural qualities. For instance, Amyot describes an approach where alternative architectures can be evaluated on the complexity and cost of the resulting message exchanges [9] as in the UCM example in Section II. Wu and Kelly have explored architecture evaluation from a security angle. They proposed a negative scenario framework where they explore "deviations" of UCM scenarios as potential security issues that impact architectural design decisions. Similar work was done by Karpati *et al.* [112], who introduced Misuse Case Maps as a modeling technique that is the anti-behavioral complement to UCM, which is used to visualize how cyber attacks are performed in an architectural context. The work of Folmer *et al.* [70] focuses on the use of scenarios (described with UCM and other means) for evaluating the usability of architectures before their implementation.

I. System Comprehension and Evolution

URN models are not just useful in a forward engineering development cycle. They can also be used in reverse-engineering, program comprehension, and evolution contexts to describe existing systems, architectures, and services.

Amyot *et al.* have proposed a static approach to recovering UCM scenarios from code, based on a manual tagging approach and a commercial tool [19]. A dynamic approach was explored by Hamou-Lhadj *et al.*, where execution traces are transformed to UCM scenarios [83]. One key step is the identification of utility functions in the code, which can be eliminated in order to shorten the resulting scenarios without loss of understandability.

Hewitt and Rilling proposed a lightweight approach to identify the impact of requirement changes on a system based on the dependencies and potential ripple effects that can be inferred from a UCM model [94]. Shiri's thesis expanded on this work with UCM-based techniques for impact analysis at the requirements level, prediction of regression testing effort, and feature interaction analysis, in order to support system evolution activities [173].

In his thesis, Störmer proposed the Software Quality Attribute Analysis by Architecture Reconstruction (SQUA3RE) method, where architectures are recovered based on a combination of UCM scenarios and time-performance models [179]. The UCM models are built manually based on interviews. His study highlights that the participants appreciated the intuitiveness of UCM for showing flows of events and mappings to architectural components, and for decomposing structure and behavior. Störmer developed his own UCM tool, called Architecture Explorer, with support for timestamps and response-time requirements similar to Scratchley's [169].

More recently, Díaz-Pace *et al.* [66] presented an approach called ArchSync (supported by an Eclipse-based UCM tool with the same name and initially developed by Blech) that helps architects synchronize architectural documentation expressed through UCM with Java source code, as modifications are being made on the code. Execution traces are used as an input, and inconsistencies with the architectural UCM model are then highlighted. ArchSync is actually complementing another tool (FLABot), discussed by Soria *et al.* [178]. FLABot is a fault-localization tool that uses a UCM specification and a set of architecture-to-code mappings in order to guide the architect in the identification of code regions with possible faults. UCMs were used as they "fit well with the exploration of cause-effect paths for faults".

Note that an interesting study by Ölvingson *et al.* evaluated UCM as a requirements engineering and system comprehension technique for the development of information systems in inter-organizational public health settings [149]. The UCM notation was found to be at a suitable level of abstraction and useful in generating intuitive requirements. At the time (2002), the authors also identified the absence of guidelines on how to use the notation as well as the difficulty in distinguishing as-is models from to-be models as weaknesses that could benefit from further attention.

J. Testing and Verification

The availability of scenarios in URN models makes URN attractive for requirements-based testing. Beyond the various analysis techniques discussed so far in the sections on formalization, transformations, and feature interactions, we distinguish three main categories of approaches for the generation of test purposes from UCM models [23].

The first category is based on the usage of UCM models as is. For example, Amyot's thesis defines a collection of testing patterns that can be used to manually cover a UCM model [8]. Charfi's thesis also proposes an approach that generates test goals (this time, as LOTOS processes) by automatically covering the paths in UCM models [59]. Feng and Lee take into consideration statistical usage at the UCM level in order to guide the selection of important test cases for frequent paths [69].

The second category exploits standard UCM scenario definitions and path traversal algorithms. The techniques used for generating MSC scenarios can be reused as is to generate test purposes in MSC or in other formats. For example, Amyot *et al.* have used scenario definitions and the UCMNAV tool to generate test cases automatically for a Web application [22].

The third category requires the transformation of the UCM model to a formal specification from which existing test generation methods can be used. All of the mappings discussed in the formalization section are hence useful in this context.

In order to turn test purposes extracted from UCM models into executable test cases, several issues must be addressed. For instance: UCM models do not include domain data, implementation messages and interfaces are unknown as UCM abstract from inter-component communication, and unfeasible test purposes might be selected depending on the chosen coverage strategy. While these concepts currently are not first-class URN modeling entities, some of them might be modeled indirectly with URN metadata and links.

Although Jaskó *et al.* suggest that GRL can be used to provide rationale for test purposes and test strategies [108], Arnold *et al.* observe that URN in general should be augmented with a testable model for functional and non-functional requirements, an implementation under test, and explicit bindings between the two views [29]. Through their experience in developing a model-based testing environment for .NET applications, they show the feasibility of having a URN-based testing approach where a URN model can be transformed into a testable requirements model from which executable test cases can be generated and then tested against an instrumented implementation (from which additional information is generated at run-time to check compliance with non-functional requirements).

Hassine *et al.* take a different angle and consider UCM as a property specification language rather than a source of test purposes [91]. The resulting pattern system is mapped to popular temporal logics such as CTL, TCTL and ArTCTL (architectural real-time temporal logic, an extension to TCTL that provides temporal logics with architectural scopes). Properties extracted from UCM models can then be verified against designs and implementations using model checkers.

K. Patterns

As anticipated by Buhr in the 90's [50], URN had a positive impact on the pattern-oriented development community. URN also benefited directly from some work in that community as well.

In her thesis, Andrade developed a substantial UCM-based pattern language to describe common aspects of mobile telecommunication systems [25], whereas Billard used UCM to describe patterns of interactions in agent systems [34].

How to create UCM scenario models and exploit them are also the topics of several contributions. For example, Mussbacher and Amyot proposed a collection of UCM modeling patterns for describing and composing telecommunication features [135], whereas Amyot described a pattern language to derive test purposes from UCM models [8].

UCM can help describe the solution space of patterns, but GRL can also capture the various forces at play. This had already been observed for other goal-oriented languages such as the NFR framework, used by Gross and Yu to document pattern forces [81]. The GRL-UCM combination was exploited by Weiss in the description of various patterns for agent systems [188] and for Web applications [189].

GRL strategies can additionally be used to assess the qualitative impact of various solutions to a functional goal, in context, enabling users to select appropriate solutions. UCM level solutions can also be linked to each other with a proper use of stubs and plug-ins. This is at the basis of the URN-based formalization of patterns done by Mussbacher *et al.* and illustrated with an architectural pattern language [140]. Rather than using GRL strategies, Weiss and Mouratidis provided a mapping from GRL to Prolog to perform an evaluation of goal models describing the trade-offs that exist when selecting amongst a number of security patterns [193].

More recently, Behnam *et al.* used URN to formalize a pattern-based framework for goal-driven business process modeling, which can be used to derive suitable business processes (traceable to its objectives) for an organization whose context is also formalized with GRL. They illustrated the framework with a healthcare example. Pourshahid *et al.* [158] also explored business process reengineering patterns that require the combination of goals, scenarios (for processes), aspects, and indicators, as supported by the Aspect-oriented URN notation [134]. They demonstrated the potential of such patterns for evolving business processes at run-time.

URN was also influenced by the literature on workflow patterns [162], which led directly to the introduction of new types of UCM stubs (synchronization and blocking) in the standard. The resulting expressiveness of UCM is compared to other scenario languages (i.e., BPMN, UML activity diagrams, and BPEL4WS) with the help of 43 workflow patterns in [136].

L. User Interface Engineering

In our literature survey, we detected an original use of URN in the domain of user interface engineering, which was unforeseen when the standardization work was initiated.

Folmer *et al.* [70] proposed a scenario-based usability assessment method to evaluate whether a given software architecture meets its usability requirements. Their Scenario-based Architecture Level UsabiliTy Assessment (SALUTA) makes use of UCM in its scenario evaluation step. One research issue they identified is the need for UCM to express static properties of usability.

Such properties are actually proposed as UCM extensions in Alsumait's thesis [6]. Alsumait *et al.* started investigating UCM as a medium for integrating task analysis (a topic already explored by Lethbridge and Singer [122]) and usability into a user-centered requirements engineering process [7]. The UCM notation was extended with concepts for supporting tasks, dialogs, and grouping/layout of user interface elements. They observed the potential of their extended UCM models for capturing user interface requirements. van der Poll *et al.* extended this work to provide a Z-based interpretation of UCM models, which enables formal usability analysis [183]. They provided an e-mail system as an example.

Alsumait's Scenario and Use Case-based for Requirements Engineering (SUCRE) framework provides the latest details on this work. The extended UCM notation for user interface modeling is presented together with examples as well as a set of analysis techniques based on metrics and on mappings to Z and LOTOS.

These extensions were not included in the URN standard because usability engineering was not one of the original objectives. In addition, the extensibility of URN in terms of metadata and links enables the support of most of the extensions proposed here, except for their graphical representation.

M. E-Business and Business Process Modeling

For more than a decade, UCM have been used in the design of e-business applications. In particular, Gordijn and Akkermans, with the help of de Bruin and van Vliet, defined UCM extensions and ontological concepts for capturing (economic) value that became very successful over the years [77]. This work resulted in a framework for value-based requirements engineering known as *e^3-value* and detailed in Gordijn's thesis [78]. *e^3-value*, which has now become an independent language with its own tools and user community (see www.e3value.com), is used to model organizations in a value web, exchanging things of economic value with each other. Relationships between *e^3-value* and goal modeling à la GRL are explored in [184].

Rather than focusing on value exchanges, Lethbridge and Singer [122] used UCM as one of their techniques for representing the work (i.e., the processes) of software engineers after observing it through shadowing. Later, Bleinstein *et al.* [36] proposed to use GRL combined with Jackson's problem frames [107] and role activity diagrams in a requirements engineering approach that captures both business strategy and process requirements for e-business systems. In their approach, projections (rather than URN links) are used to connect the views. Additional work focusing on business process alignment was done in [37].

The combination of GRL and UCM for describing business objectives and processes/workflows then became very apparent. Weiss and Amyot argued that URN is a suitable notation for business process modeling, business evolution, and business alignment, and they illustrated their case with a supply chain management example [190]. Pourshahid and Tran have also shown the usefulness of URN in the modeling and analysis of trust in e-commerce systems [159].

In order to better handle business management concepts and be able to capture quantities in terms of domain-specific units, Pourshahid and Chen have extended GRL with the concept of Key Performance Indicators (KPI) [157]. A KPI converts a value observed in a running business process or context to a satisfaction level in the [-100,100] range understood by GRL. Target, threshold, and worst-case values are defined in each KPI to assist this conversion. In jUCMNav, GRL strategy definitions were also extended to access external sources of information (e.g., data warehouses, sensors, business intelligence application, or performance management tools) for online monitoring, management, and runtime adaptation of business processes [60]. These extensions were used in methodologies applied to several real healthcare process examples by, e.g., Pourshahid [156] and Kuziemsky *et al.* [119].

N. URN Tools

Several tools have been developed over the years to support GRL, UCM, or URN modeling. On the UCM side, the development of the C++, multi-platform UCMNav tool [181] ended around 2005 in favor of the new Eclipse-based jUCMNav [110][137], a Java tool that actually started as an undergraduate student project [117]. jUCMNav (see Figure 4) was originally a simple UCM tool that prevents the creation of syntactically incorrect URN models. As part of his thesis [161], Roy added a GRL editor and invented the concept of GRL strategy, supported by a hybrid propagation algorithm with color feedback, as seen in Figure 1 [160]. He also provided goal-scenario traceability management (via URN links) and support for GRL catalogues. Kealey, in his thesis [114], implemented a flexible UCM traversal mechanism with color highlight (see Figure 2), together with an MSC export feature [115]. Kealey developed a mechanism where GRL evaluation results can influence the traversal of UCM paths, and vice-versa. Numerous semantic variation points in UCM were also identified for further clarification. Many of the contributions by Roy and Kealey found their way into the URN standard. Yan added a mechanism for the verification of user-defined static semantic rules and constraints written in OCL [24], whereas Gao recently contributed support for the import and export of URN models in the XML-based standard interchange format [71]. Other features, some of which are discussed in the previous sections, from dozens of contributors are also present in this tool.

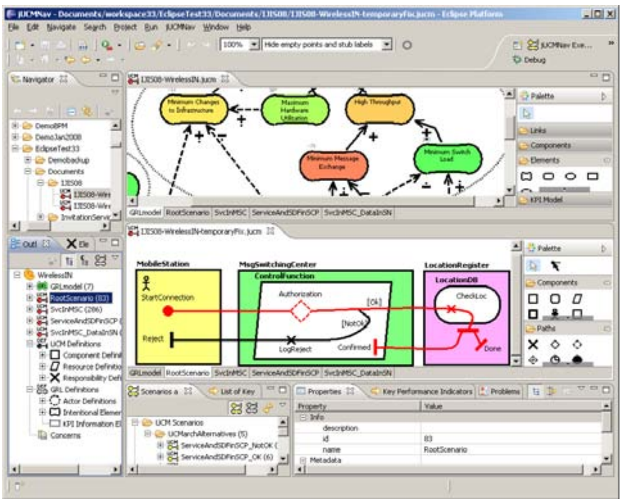


Figure 4 Overview of the jUCMNav tool interface.

jUCMNav is for the moment the only tool that supports both goal and scenario modeling and analysis. As discussed earlier, other tools with partial and specialized support for UCM also exist: Störmer's Architecture Explorer for architecture recovery activities [179] and ArchSync for the documentation, maintenance and diagnosis of applications written in Java [66].

On the GRL side, Liu extended Yu's Organization Modelling Environment (OME 3) to support an early version of the notation and an interactive propagation mechanism [198]. This was the only GRL tool available for a long time, and it helped shape GRL as it is known today. OME 3 was also deprecated in favor of an Eclipse-based version called OpenOME [150], a project led by Yu whose major contributors include Ernst, Horkoff, Ng, Olinescu, and Y. Yu. OpenOME integrates with other platforms (such as Protégé and Visio) to support goal-oriented and agent-oriented modeling. Strategy-based evaluation is not supported, but there are a variety of analysis features, including interactive propagation.

There also exists a Visio-based tool, named Sandrila SDL [163], which supports GRL modeling, without analysis capabilities. Other goal-oriented modeling tools are discussed and evaluated on the *i** Wiki [97].

O. Requirements Management and Compliance

URN models capture only a fraction of the requirements of telecommunication standards and software products. Accordingly, such models need to be used in cooperation with complementary general requirements, and both views must be linked in a way that supports traceability, navigation, and analysis. The proposed URN standard ensures that model elements are uniquely identifiable inside a specification, which helps supporting such links. However, one important challenge that remains is the maintenance of these links as models and general requirements evolve.

Jiang proposed an approach to export UCM scenario models to the IBM Rational DOORS requirements management system and to maintain relationships as both views evolve over time [109]. Originally developed for UCMNAV, this functionality is now supported as an export filter for jUCMNav, thanks to the efforts of Kim *et al.* [116]. The tool also provides a link auto-completion mechanism to minimize the possibly large number of links that have to be created manually by DOORS users between external and UCM requirements. Roy later extended this mechanism to support GRL and URN links [161].

Ghanavati built on this work to study the compliance of organization goals and business processes against laws and policies [73]. URN is used both to capture the goals and processes of the organization and to model legislation. Exporting and linking both views to DOORS enables one to assess the legal compliance of business processes, as well as maintain it when laws or processes evolve [74]. Her original framework was extended to exploit contributions of processes to the elements of the law/policy model, enabling the measure of partial compliance [76].

All of the above contributions convinced us that URN models can indeed be combined with other types of requirements and design artifacts in a requirements management context.

P. Aspect-oriented Modeling

Over the last decade, aspect-oriented modeling (AOM) techniques have been developed for many requirements and design notations in order to better address separation of concern issues found in complex models.

The UCM notation's ability to model aspects was identified in the late 90's by Buhr [48] but received little attention since then with the exception of work by de Bruin and van Vliet [65]. The top-down approach by de Bruin and van Vliet explicitly added a "Pre" stub and a "Post" stub for each location on a map that requires a change. The stubs allowed behavior to be added before or after the location by plugging refinement maps into the stubs.

In 2005, work started on the Aspect-oriented User Requirements Notation (AoURN), which is best described in Mussbacher's thesis [134]. Mussbacher proposed aspect-oriented extensions to UCM and GRL models to unify goal-oriented, scenario-based, and aspect-oriented techniques in one modeling framework. AoURN allows a concern to be encapsulated even if it is crosscutting other concerns, thus leading to improvements in the modularity, maintainability, and reusability of URN models. The concept of a concern was deemed important enough to be included in the URN standard. In AoURN, patterns and composition rules are described with URN itself, thus allowing for a flexible and exhaustive approach that is not limited to a particular composition language but can harness the full expressive power of URN [141]. In AOM, patterns specify where an aspect is to be applied, and composition rules specify how an aspect is to be applied at the location identified by a pattern. The matching and composition mechanism of AoURN goes well beyond typical composition operators and includes among others concurrent, loop, and interleaved composition. AoURN's mechanism is further enhanced by taking semantic equivalences of URN related to hierarchical structuring into account [142]. This approach allows common refactoring operations to be performed on an AoURN model without breaking aspectual specifications.

AoURN has been used for various applications and some of them are discussed here as examples. A large challenge problem posed by the aspect-oriented modeling community involved a safety-critical, reactive system [139]. In the context of business process monitoring and improvement, AoURN enabled changes to business processes based on business process redesign patterns and an assessment of the shortcomings of the current business process with the help of goal models [158]. Crosscutting concerns were also described with AoURN for a SOA-based application and added to composite services based on an assessment of non-functional properties modeled with URN [31]. Finally, AoURN has also been applied to model commonalities and variabilities in Software Product Lines [138].

Q. Support for Standardization

Several authors have emphasized the role of URN in the development of standards and systems, beyond the vision described by Hodges and Visser in 1999 [96]. For instance, Sales described how UCM can fit in the development of IETF protocol standards [167]. Adamis *et al.* briefly compared ITU-T languages (including URN) with UML, and illustrated how the former can be used together to model systems [4]. Medve also studied the ITU-T languages (with an emphasis on UCM) and UML, this time in the context of system re-engineering [128].

From the perspective of ITU-T standardization processes, URN's capability to model goals and scenarios fits well the so-called "stage 1" requirements descriptions described in Recommendation I.130 [100]. In the telecommunications networks management domain, Recommendation M.3020 proposes the description of various types of requirements (functional, non-functional, administrative, etc.) with textual use case and UML use case diagrams [101]. Again, URN models fit nicely in such a process as they bring formality and executability to the use cases while enabling concrete support for goal models, which are useful to derive and analyze non-functional and administrative requirements.

More recently, the growing interest in standards for Next-Generation Networks (NGN) brought new needs for improved service description and engineering approaches. Ideally, one would like to specify and analyze services and standards at a high level of abstraction, using modeling concepts close to the user and problem domain rather than at the platform and implementation domain, and then be able to derive design components and implementations from service models with a high degree of automation. This is essentially the abstraction level targeted by URN, as discussed in [12]. GRL goal models offer a holistic view that integrates stakeholder goals, non-functional requirements, and alternative operational solutions for design time decisions, supplemented with indicators that enable adaptive behavior at runtime. UCM offer scenarios that express variability points explicitly while offering much flexibility in ordering activities, which may be bound to components or not. The integration of GRL and UCM, combined with strategies and scenario definitions, and possibly with aspect-oriented concepts, emphasizes the importance of enabling dynamic choices in the service modeling and design phases in order to take into account contextual information and differentiated service availability requirements in dynamic service composition, which are key aspects of NGN services.

VI. The Next Ten Years

The previous section proved that much development related to URN has happened in the past ten years. Yet, we expect the next ten years to be even more exciting in terms of the diversity of application domains for URN, and of new modeling and analysis features that will be emerging. In particular, we predict major developments in the following eight areas.

A. Domain-Specific Profiles

There will be a need to tailor and extend URN for specific application domains. The need for extensions was already raised on several occasions in the previous section, for instance in the areas of user interface engineering and testing. It has been observed for other domains as well, including the popular area of software product lines [44], or emerging domains like home network systems [151].

The URN standard already offers mechanisms that, when combined, support the *profiling* of the language to a particular domain. 1) Metadata are name-value pairs that can be used to tag any URN element, similar to stereotypes in UML. 2) URN links can be used to specify relationships between any pair of URN elements. 3) URN concerns can be used to group any collection of URN elements (including other concerns). 4) OCL constraints can be defined to restrict the use of the language or of its extensions [24]. For example, a URN profile for *i** is defined and implemented in [18], demonstrating that URN can represent concepts and constraints found in another language.

ITU-T is also inviting contributions on the definition of UML profiles for all its languages. Abid *et al.* have already defined a tool-supported UML profile for GRL [3] (based on ITU-T guidelines). Molina *et al.* also proposed a UML profile for measurable goal modeling (i.e., with indicators), with the GRL syntax as its concrete notation [131]. However, much work remains to be done to cover URN entirely. Progress in this direction may also cause the URN standard to include the capability of creating profiles (including changing the shapes of some elements) as first-class entities.

B. Enhanced Workflow Executions

The URN standard defines advanced workflow operators (e.g., blocking stubs with threshold and replication factors) together with the corresponding traversal rules. However, at this time, no tool is currently supporting them at the analysis/simulation level.

URN is also missing important constructs to specify properly cancellation scenarios and situations akin to exception handling. Mussbacher proposed the addition of failure points and failure start points, which would enable the concise representation and analysis of cancellation situations [134]. This topic was actually adopted in a revised version of Recommendation Z.150 in February 2011.

These new operators will require proper tool support to be useful. In addition, tools could consider supporting other execution modes like debuggers, which would be useful during the analysis of workflow models.

C. Performance Management

The concept of indicator (KPI) is another addition being considered in the short term for the URN standard and recently adopted in the revised Recommendation Z.150. We have had over three years of experience with KPI in jUCMNav and numerous models [60][157], and they have quickly become an essential part of any description of business process or adaptive system in a performance management context.

Dynamic adaptation of systems based on KPI is an exciting area of research that is expected to benefit from URN's simultaneous support for goal modeling and scenario modeling. With KPI, real-world values may influence the evaluation of goal models and drive the simulation of what-if scenarios to guide and inform dynamic adaptation at run-time.

There might however be a need for a more flexible definition of what a KPI is at the metamodel level. For example, the mapping of an observed value to a GRL satisfaction level could be done differently than the current simple linear correspondence. KPI in a model could also influence or contribute to each other, enabling the computation of aggregate KPI. Trends computed from external data sources could also be integrated in GRL models. Such improved KPI definitions are actually being explored in the Business Intelligence Model language [30], inspired partly from URN.

D. Compliance Management

The assessment of compliance of business goals and processes with laws and policies is also a domain where URN is expected to have an impact in the next ten years.

Following the initial work of Ghanavati *et al.* [76], there are still important challenges to be addressed, including the systematic extraction of URN models from textual laws and policies, and the prioritization of efforts to improve a partial degree of compliance. Other research questions include the potential need for deontic modalities (e.g., obligations, permissions, and interdictions) or Hohfeldian classes of rights in URN models of laws and policies [75], as well as the role of indicators for measuring compliance [172].

E. Formal Semantics

URN has a formal description of its abstract syntax, but only a natural language description of its semantics in terms of rules constraining the propagation in GRL and the traversal of UCM paths. There is a need to reduce the number of semantic variation points (known and unknown) in the language. This could be done with a complete mapping to an underlying formalism. Many partial mappings were discussed in Section V.D, but complete mappings are much more difficult to achieve, especially if they are to span GRL and UCM. There might also be a way to provide a formal description in the form of a virtual machine describing the interpretation of URN models.

F. Improved Analysis Techniques

There are many opportunities to add to the set of analysis techniques currently used in URN. For example, the current GRL propagation algorithms based on strategies are mainly bottom-up, in a way similar to test cases. The availability of top-down algorithms would provide substantial benefits to many users, who would simply ask the model how to optimize one or several actors or intentional elements given an initial context. The main difficulty is usually that top-down algorithms correspond to a kind of search problem and are hence much more complex than bottom-up algorithms. Weiss and Mouratidis [193] have a mapping from GRL to Prolog that might help solve this issue. Okamura *et al.* [151] may also have elements of answers.

The analysis of aspect-oriented GRL and UCM in AoURN is still an open issue [134], and this may even have an impact on how strategies and scenarios are defined in URN.

Time extensions and analysis for UCM, as proposed by Hassine *et al* [85], are also relevant to URN and hence deserve some attention.

Finally, from an analysis perspective, there is a need for a tighter integration between GRL and UCM, which could result in the combination of strategies and scenarios in one logical unit in the URN standard.

G. Improved Model Representations

Recent analyses of the graphical syntaxes of UCM by Genon *et al.* [72] and of *i** by Moody *et al.* [133] reveal that there are many problems with the cognitive fitness of the symbols used, and with the completeness of the language's concrete syntax. For instance, performance annotations and stub bindings do not have any visual representation in URN. The concrete visual syntaxes could be reviewed and completed in the standard. Additional concrete syntaxes, such as textual or tabular for GRL graphs, also deserve to be explored.

H. Guidelines and Methodologies

Last but not least, there is currently a lack of guidelines and methodologies for URN modeling, both in isolation and in combination with other modeling languages. This was already raised as an issue for UCM in 2002 by Ölvingson *et al.* [149], and this is unfortunately still true to some extent today. A related challenge is the automated or semi-automated transformation of URN models to other modeling techniques. While many transformations have already been investigated in Section V.E, there is a need to revisit some of these transformations and new transformations to emerging modeling techniques in the light of recent technologies and applications such as AOM, domain-specific languages, dynamic adaptation of systems, and SOA-based systems. ITU-T also invites contributions to the definition of a URN-based methodology. Better guidelines and methodologies could have a strong impact on the adoption of URN in industry.

VII. CONCLUSIONS

This paper reports on a systematic literature survey about the development of the User Requirements Notation. Section II first gives an overview of the notation and typical analysis techniques, followed by a discussion of the origins of URN in the 1990's. In Section IV, the analysis of the 281 papers selected for this study reveals that URN is a growing, global language, both in terms of contributors and users.

In section V, we introduce and commented on 17 categories of contributions to URN during its first ten years. This period is mainly characterized as follows:

- Simple formalization of URN in terms of abstract and concrete syntaxes;
- The definition of analysis techniques such as GRL strategies with forward propagation and UCM scenario definitions with a path traversal mechanism;
- The completion of a first version of the URN standard;
- Simple combinations of GRL and UCM in models, with traceability, completeness, and consistency analysis, and with GRL strategies and UCM scenarios that can influence each others;
- Emerging techniques for the analysis of feature interactions, performance, and architectures;
- The availability of open-source, Eclipse-based tool support (jUCMNav);
- A multitude of application domains explored, mainly related to telecommunication systems and reactive systems at the beginning, and later mainly related to business processes and aspect-oriented modeling;
- Many formalizations and transformations also explored.

We believe that the next ten years of URN development will be even more active than the first ten and will focus on the major topics identified in Section VI, including domain-specific profiles, enhanced workflow executions, performance and compliance management, formal semantics, improved analysis techniques and model representations, and guidelines and methodologies.

We hope this survey paper represents a useful one-stop document for URN beginners and experts alike. We also take the opportunity to invite users and other interested parties to get actively involved in future developments of the User Requirements Notation.

ACKNOWLEDGMENT

The authors wish to thank the many people who have contributed to the success of the User Requirements Notation over the years, with special thanks to C.M. Woodside and T. Gray for comments on an earlier draft of this paper. This work was supported in part by the Discovery grants and Postgraduate Scholarships programs from NSERC (Canada) and by the Ontario Graduate Scholarship Program.

REFERENCES

[1] T. Abdelaziz, *Towards a Comprehensive Agent-Oriented Software Engineering Methodology*, Doctoral Dissertation, Universität Duisburg-Essen, Germany, October 2008

[2] T. Abdelaziz, M. Elammari, and R. Unland, “Visualizing a Multiagent-Based Medical Diagnosis System Using a Methodology Based on Use Case Maps”, in *MATES 2004: multiagent system technologies*, LNCS 3187, Springer, pp. 198–212, 2004. doi:10.1007/978-3-540-30082-3_15

[3] 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, 14th Int. SDL Forum*, LNCS 5719, Springer, pp. 133–148, September 2009. doi:10.1007/978-3-642-04554-7_9

[4] G. Adamis, R. Horváth, Z. Pap, and K. Tarnay, “Standardized languages for telecommunication systems”. *Computer Standards & Interfaces*, 27(3), Elsevier, pp. 191–205, March 2005. doi:10.1016/j.csi.2004.09.005

[5] C.P. Ayala, C. Cares, J.P. Carvallo, G. Grau, M. Haya, G. Salazar, X. Franch, E. Mayol, and C. Quer, “A Comparative Analysis of i*-Based Goal-Oriented Modelling Languages”, in *Int. Workshop on Agent-Oriented Software Development Methodologies (AOSDM @SEKE)*, Taipei, China, pp. 43–50, July 2005

[6] A. Alsumait, *User Interface Requirements Engineering: A Scenario-Based Framework*. Ph.D. thesis, Concordia University, Canada, August 2004.

[7] A. Alsumait, A. Seffah, and T. Radhakrishnan, “Use Case Maps: A Roadmap for Usability and Software Integrated Specification”, in 17th World Computer Congress - TC13 Stream on Usability, IFIP, pp. 119–131, August 2002,

[8] D. Amyot, *Specification and Validation of Telecommunications Systems with Use Case Maps and LOTOS*. Ph.D. thesis, SITE, University of Ottawa, Canada, Sept. 2001.

[9] D. Amyot, “Introduction to the User Requirements Notation: Learning by Example”. *Computer Networks*, 42(3), pp. 285–301, June 2003. doi:10.1016/S1389-1286(03)00244-5

[10] D. Amyot and R. Andrade, “Description of Wireless Intelligent Network Services with Use Case Maps”, in *17th Brazilian Symposium on Computer Networks (SBRC'99)*, Salvador, Brazil, pp. 418–433, May 1999.

[11] D. Amyot, R. Andrade, L. Logrippo, J. Sincennes, and Z. Yi, “Formal Methods for Mobility Standards”, in *IEEE 1999 Emerging Technology Symposium on Wireless Communications & Systems*, Dallas, USA, pp. 14.1–14.7, April 1999. doi:10.1109/ETWCS.1999.897332

[12] D. Amyot, H. Becha, R. Bræk, and J.E.Y. Rossebø, “Next Generation Service Engineering”, in *ITU-T Innovations in NGN - Kaleidoscope Academic Conference*, Geneva, Switzerland, pp. 195–202, May 2008. doi:10.1109/KINGN.2008.4542266

[13] D. Amyot, F. Bordeleau, R.J.A. Buhr, and L. Logrippo, “Formal support for design techniques: a Timethreads-LOTOS approach”, in *FORTE VIII, 8th Int. Conf. on Formal Description Techniques*, Chapman & Hall, pp. 57–72, 1995.

[14] D. Amyot, L. Charfi, N. Gorse, T. Gray, L. Logrippo, J. Sincennes, B. Stepien, and T. Ware, “Feature Description and Feature Interaction Analysis with Use Case Maps and LOTOS”, in *Sixth International Workshop on Feature Interactions in Telecommunications and Software Systems (FIW'00)*, IOS Press, pp. 274–289, May 2000.

[15] D. Amyot, D.Y. Cho, X. He, and Y. He, “Generating Scenarios from Use Case Map Specifications”, in *Third Int. Conf. on Quality Software (QSIC'03)*, IEEE CS, pp. 108–115, Nov. 2003. doi:10.1109/QSIC.2003.1319092

[16] D. Amyot and A. Eberlein, “An Evaluation of Scenario Notations and Construction Approaches for Telecommunication”. *Telecommunications Systems Journal*, 24(1), Kluwer, pp. 61–94, September 2003. doi:10.1023/A:1025890110119

[17] D. Amyot, S. Ghanavati, J. Horkoff, G. Mussbacher, L. Peyton, and E. Yu, “Evaluating Goal Models within the Goal-oriented Requirement Language”. *Int. Journal of Intelligent Systems*, 25(8), Wiley, pp. 841–877, August 2010. doi:10.1002/int.20433

[18] D. Amyot, J. Horkoff, D. Gross, and G. Mussbacher, “A Lightweight GRL Profile for i* Modeling”, in *3rd Int. Work. on Requirements, Intentions and Goals in Conceptual Modeling (RIGiM 2009)*, LNCS 5833, Springer, pp. 254–264, Nov. 2009. doi:10.1002/int.20433

[19] D. Amyot, M. Mussbacher, and N. Mansurov, “Understanding Existing Software with Use Case Map Scenarios”, in *3rd SDL and MSC Workshop (SAM02)*, LNCS 2599, Springer, pp. 124–140, June 2002. doi:10.1007/3-540-36573-7_9

[20] D. Amyot and L. Logrippo, “Use Case Maps and LOTOS for the Prototyping and Validation of a Mobile Group Call System”. *Computer Communications*, 23(12), pp. 1135–1157, July 2000. doi:10.1016/S0140-3664(99)00242-X

[21] D. Amyot, L. Logrippo, R.J.A. Buhr, and T. Gray, “Use Case Maps for the Capture and Validation of Distributed Systems Requirements”, in *Fourth Int. Symposium on Requirements Engineering (RE'99)*, IEEE CS, pp. 44–53, June 1999. doi:10.1109/ISRE.1999.777984

[22] D. Amyot, J.-F. Roy, and M. Weiss, “UCM-Driven Testing of Web Applications”, in *12th SDL Forum (SDL 2005)*, LNCS 3530, Springer, pp. 247–264, June 2005. doi:10.1007/11506843_18

[23] D. Amyot, M. Weiss, and L. Logrippo, “Generation of Test Purposes from Use Case Maps”. *Computer*

Networks, 49(5), Elsevier, pp. 643–660, December 2005. doi:10.1016/j.comnet.2005.05.006

[24] D. Amyot and J.B. Yan, "Flexible Verification of User-Defined Semantic Constraints in Modelling Tools", in *18th Int. Conf. of Computer Science and Software Engineering (CASCON)*, IBM CAS, October 2008. doi:10.1145/1463788.1463798

[25] R. Andrade, *Capture, Reuse, and Validation of Requirements and Analysis Patterns for Mobile Systems*. Ph.D. thesis, SITE, Univ. of Ottawa, Canada, May 2001.

[26] R. Andrade and L. Logrippo, "Reusability at the Early Development Stages of the Mobile Wireless Communication Systems", in *4th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2000)*, IIIS, Orlando, USA, pp. 11–16, July 2000.

[27] R. Andrade, W. Viana, and D.P. Menezes, "A high-level application framework for mobile system development: IMT-2000 case study", in *9th Int. Conf. on Telecommunications*, IEEE, Beijing, China, pp. 321–325, June 2002.

[28] P. Araya and H. Antillanca, "Una metodología de agents", in *1er. Workshop Chileno de Ingeniería de Software*, Punta Arenas, Chile, November 2001.

[29] D. Arnold, J.-P. Corriveau, and W. Shi, "Scenario-Based Validation: Beyond the User Requirements Notation", in *21st Australian Software Engineering Conf. (ASWEC 2010)*, IEEE CS, pp 75–84, April 2010. doi:10.1109/ ASWEC.2010.29

[30] D. Barone, E. Yu, J. Won, L. Jiang, and J. Mylopoulos, "Enterprise Modeling for Business Intelligence", in *The Practice of Enterprise Modeling*, LNBIP 68, Springer, pp. 31–45, 2010. doi:10.1007/978-3-642-16782-9_3

[31] H. Becha, G. Mussbacher, and D. Amyot, "Modeling and Analyzing Non-Functional Requirements in Service Oriented Architecture with the User Requirements Notation". *Non-functional Properties in Service Oriented Architecture: Requirements, Models and Methods*, IGI Global, USA, pp. 48–72, 2011. doi:10.4018/978-1-60566-794-2.ch003

[32] S.A. Behnam, D. Amyot, and G. Mussbacher, "Towards a Pattern-Based Framework for Goal-Driven Business Process Modeling", in *8th Int. Conf. on Software Engineering Research, Management and Applications (SERA2010)*, IEEE CS, pp. 137–145, May 2010. doi:10.1109/SERA.2010.27

[33] E.A. Billard, "Operating system scenarios as Use Case Maps", in *Fourth Int. Work. on Software and Performance (WOSP 2004)*, ACM Press, pp. 266–277, January 2004. doi:10.1145/974044.974087

[34] E.A. Billard, "Patterns of agent interaction scenarios as Use Case Maps". *IEEE Transactions on Systems, Man and Cybernetics*, 24B:4, pp. 1933–1939, August 2004. doi:10.1109/TSMCB.2004.828192

[35] G. Birkhoff, *Lattice theory*. American Mathematical Society, 1967.

[36] S.J. Bleistein, K. Cox, and J. Verner, "Requirements Engineering for e-Business Systems: Integrating Jackson Problem Diagrams with Goal Modeling and BPM", in *11th Asia Pacific Software Engineering Conference (APSEC 2004)*, IEEE CS, pp. 410–417, November 2004. doi:10.1109/APSEC.2004.84

[37] S.J. Bleistein, K., Cox, J. Verner, and K.T. Phalp, "Requirements engineering for e-business advantage", *Requirements Engineering*, 11(1), pp. 4–16, March 2006. doi:10.1007/s00766-005-0012-7

[38] E. Börger and R. Stärk, *Abstract State Machines: A Method for High-Level System Design and Analysis*. Springer-Verlag, 2003.

[39] F. Bordeleau, *A Systematic and Traceable Progression from Scenario Models to Communicating Hierarchical State Machines*. Ph.D. thesis, SCE Dept., Carleton University, Canada, December 1999.

[40] F. Bordeleau and R.J.A. Buhr, "The UCM-ROOM Design Method: from Use Case Maps to Communicating State Machines", in *Conf. on the Engineering of Computer-Based Systems (ECBS)*, pp. 167–179, March 1997. doi:10.1109/ECBS.1997.581850

[41] F. Bordeleau and D. Cameron, "On the Relationship between Use Case Maps and Message Sequence Charts", in *2nd Workshop on SDL and MSC (SAM 2000*), Grenoble, France, pp. 123–138, June 2000.

[42] F. Bordeleau, J.-P. Corriveau, and B. Selic, "A Scenario-Based Approach to Hierarchical State Machine Design", in *ISORC 2000: 3rd IEEE Int. Symp. on Object-Oriented Real-time distributed Computing*, IEEE CS, pp. 78–85, March 2000. doi:10.1109/ISORC.2000.839514

[43] P. Brereton, B.A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software engineering domain". *J. of Systems and Software*, 80(4), pp. 571–583, April 2007. doi:10.1016/j.jss.2006.07.009

[44] J. Brown, R. Gawley, I. Spence, P. Kilpatrick, C. Gillan, and R Bashroush, "Requirements Modelling and Design Notations for Software Product Lines", in *First Int. Workshop on Variability Modelling of Software-intensive Systems* (VaMoS), Limerick, Ireland, January 2007.

[45] H. de Bruin, "A Grey-Box Approach to Component Composition", in *Generative and Component-Based Software Engineering (GCSE 99)*, pp. 195–209, 1999.

[46] R.J.A. Buhr, "Use Case Maps for Attributing Behaviour to System Architecture", in *Fourth Int. Work. on Parallel and Distributed Real Time Systems (WPDRTS)*, pp. 3–10, 1996.

[47] R.J.A. Buhr, "Design Patterns at Different Scales", in *Pattern Languages of Programs (PLoP96)*, June 1996.

[48] R.J.A. Buhr, "A Possible Design Notation for Aspect Oriented Programming", in *ECOOP Workshop on Aspect Oriented Programming*, Brussels, Belgium, July 1998.

[49] R.J.A. Buhr, "Use Case Maps as Architectural Entities for Complex Systems". *IEEE Transactions on Software Engineering*, 24(12), pp. 1131–1155, December 1998. doi:10.1109/32.738343

[50] R.J.A. Buhr, "Understanding Macroscopic Behaviour Patterns in Object-Oriented Frameworks, with Use Case Maps (chapter 18)". *Building Application Frameworks: Object-Oriented Foundations of Framework Design*, Wiley, pp. 415–440, September 1999.

[51] R.J.A. Buhr, D. Amyot, M. Elammari, D. Quesnel, T. Gray, and S. Mankovski, "Feature-Interaction Visualization and Resolution in an Agent Environment", in *Fifth Int. Work. on Feature Interactions in Telecommunications and Software Systems (FIW'98)*, IOS Press, pp. 135–149, July 1998.

[52] R.J.A. Buhr and R.S. Casselman, *Use Case Maps for Object-Oriented Systems*. Prentice-Hall, November 1995.

[53] G. Bush, S. Cranefield, and M.K. Purvis, "The Styx agent methodology". *Information Science Discussion Paper Series*, 2001/02, University of Otago, New Zealand, 2001.

[54] Z. Cai and E.Yu, "Addressing Performance Requirements Using a Goal and Scenario-Oriented Approach", in *CAISE'02: 14th Int. Conf. on Advanced Information*

Systems Engineering, LNCS 2348, Springer, pp. 706–710, May 2002. doi:10.1007/3-540-47961-9

[55] M. Calder, M. Kolberg, E.H. Magill, and S. Reiff-Marganiec, "Feature interaction: a critical review and considered forecast". *Computer Networks*, 41, pp. 115–141, 2003.

[56] R.S. Casselman, *A Role-Based Architectural Model Applied to Object-Oriented Systems*. Master's thesis, SCE Dept., Carleton University, Canada, August 1993.

[57] H.N. Castejón Martínez, "Synthesizing State-Machine Behaviour from UML Collaborations and Use Case Maps", in *12th SDL Forum (SDL 2005)*, LNCS 3530, Springer, pp. 339–359, June 2005. doi:10.1007/11506843_24

[58] H.N. Castejón, *Collaborations in Service Engineering: Modeling, Analysis and Execution*. Ph.D. thesis, Dept. of Telematics, NTNU, Norway, November 2008.

[59] L. Charfi, *Formal Modeling and Test Generation Automation with Use Case Maps and LOTOS*. M.Sc. thesis, SITE, University of Ottawa, Canada, Feb. 2001.

[60] P. Chen, *Goal-Oriented Business Process Monitoring: An Approach based on User Requirement Notation combined with Business Intelligence and Web Services*. M.Sc. thesis, SCS Dept., Carleton University, Canada, December 2007.

[61] J. Cheng, L. Yang, Y.-J. Kuai, and D.-F. Zhang, "Non-deterministic feature interaction filtering method based on scenarios with Use Case Map" (基于用例图呼叫处理场景的不确定性冲突过滤方法), *Hunan Daxue Xuebao / Journal of Hunan University Natural Sciences*, 32(2), pp. 104–109, April 2005.

[62] L. Chung, B.A. Nixon, E. Yu, and J. Mylopoulos, *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publisher, 2000.

[63] L. Constantine and L. Lockwood, *Software for Use: A Practical Guide to the Models and Methods of User-Centered Design*. Addison-Wesley, April 1999.

[64] H. de Bruin, "Scenario-Based Analysis of Component Compositions", in *Generative and Component-Based Software Engineering (GCSE'00)*, LNCS 2177, Springer, pp. 129–146, Oct. 2000. doi:10.1007/3-540-44815-2_10

[65] H. de Bruin and H. van Vliet, "Quality-Driven Software Architecture Composition". *Journal of Systems and Software*, 66(3), Elsevier, pp. 269–284, June 2003. doi:10.1016/S0164-1212(02)00079-1

[66] J.A. Díaz-Pace, J.P. Carlino, M. Blech, A. Soria, and M.R. Campo, "Assisting the Synchronization of UCM-based Architectural Documentation with Implementation", in *IEEE/IFIP Conf. on Software Architecture and European Conference on Software Architecture (WICSA/ECSA 2009)*, IEEE CS, pp. 151–160, September 2009. doi:10.1109/WICSA.2009.5290801

[67] C. Dongmo and J. A. van der Poll, "Use Case Maps as an Aid in the Construction of a Formal Specification", in *7th Int. Workshop on Modelling, Simulation, Verification and Validation of Enterprise Information Systems (MSVVEIS-2009)*, INSTICC Press, pp. 3–13, May 2009.

[68] M. Elammari and W. Lalonde, "An agent-oriented methodology: High-level and intermediate models", in *Proc. 1st Int. Workshop on Agent-Oriented Information Systems*, Seattle, USA, June 1999.

[69] Y. Feng and L.-S. Lee, "The Importance Analysis of Use Case Map with Markov Chains". *Int. J. of Computer Science and Information Security (IJCSIS)*, 7(1), pp. 55–62, January 2010. arXiv:1002.1692v1

[70] E. Folmer, J. van Gurp, and J. Bosch, "Scenario-based Assessment of Software Architecture Usability", in *Work. on Bridging the Gaps Between Software Engineering and Human-Computer Interaction (SE-HCI)*, IFIP, pp. 61–68, May 2003.

[71] Y. Gao, *Import/Export of URN Models in Z.151 XML File Format with jUCMNav*. M.Sc. project, SITE, University of Ottawa, Canada, January 2010.

[72] N. Genon, D. Amyot, and P. Heymans, "Analysing the Cognitive Effectiveness of the UCM Visual Notation", in *6th Workshop on System Analysis and Modelling (SAM 2010)*, LNCS, Springer, October 2010 (to appear).

[73] S. Ghanavati, *A Compliance Framework for Business Processes Based on URN*. M.Sc. thesis, SYS, University of Ottawa, Canada, May 2007.

[74] S. Ghanavati, D. Amyot, and L. Peyton, "Towards a Framework for Tracking Legal Compliance in Healthcare", in *19th Int. Conf. on Advanced Information Systems Engineering (CAiSE'07)*, LNCS 44495, Springer, pp. 218–232, June 2007. doi:10.1007/978-3-540-72988-4_16

[75] S. Ghanavati, D. Amyot, A. Siena, A. Perini, and A. Susi, "Towards a Framework for Business Process Compliance", in *Int. Workshop on Goal-based Business Process Engineering (WGBP 2010)*, IEEE CS, pp. 330–334, October 2010. doi:10.1109/EDOCW.2010.46

[76] S. Ghanavati, A. Siena, D. Amyot, A. Perini, L. Peyton, and A. Susi, "Integrating Business Strategies with Requirement Models of Legal Compliance". *Int. J. of Electronic Business*, Inderscience Publishers, pp. 260–280, 2010. doi:10.1504/IJEB.2010.034171

[77] J. Gordijn and J.M. Akkermans, "Value-based Requirements Engineering Exploring Innovative e-Commerce Ideas". *Requirements Engineering*, 8(2), Springer, pp. 114–134, 2003. doi:10.1007/s00766-003-0169-x

[78] J. Gordijn, *Value-based Requirements Engineering Exploring Innovative e-Commerce Ideas*. Ph.D. thesis, Vrije Universiteit, Amsterdam, The Netherlands, June 2002. doi:10.1007/s00766-003-0169-x

[79] J. Gordijn and J.C. van Vliet, "Integral Design of E-Commerce Systems: Aligning the Business with Software Architecture through Scenarios", in *ICT-Architecture in the BeNeLux (ICT 1999)*, 1999.

[80] N. Gorse, *The Feature Interaction Problem: Automatic Filtering of Incoherences & Generation of Validation Test Suites at the Design Stage*. M.Sc. thesis, SITE, University of Ottawa, Canada, September 2001.

[81] D. Gross and E.S.K. Yu, "From Non-Functional Requirements to Design through Patterns". *Requirements Engineering*, 6(1), Springer, pp. 18–36, 2001.

[82] R. Guan, *From Requirements to Scenarios through Specifications: A Translation Procedure from Use Case Maps to LOTOS*. M.Sc. thesis, SITE, University of Ottawa, Canada, September 2002.

[83] A. Hamou-Lhadj, E. Braun, D. Amyot, and T. Lethbridge, "Recovering Behavioral Design Models from Execution Traces", in 9th *European Conf. on Software Maintenance and Reengineering (CSMR)*, IEEE CS, pp. 112–121, March 2005. doi:10.1109/CSMR.2005.46

[84] J. Hassine, *Feature Interaction Filtering and Detection with Use Case Maps and LOTOS*. M.Sc. thesis, SITE, University of Ottawa, Canada, February 2001.

[85] J. Hassine, *Formal Semantics and Verification of Use Case Maps*. Ph.D. thesis, CSCE dept., Concordia University, Canada, April 2008.

[86] J. Hassine, "AsmL-Based Concurrency Semantic Variations for Timed Use Case Maps", in *Abstract State Machines, Alloy, B and Z (ABZ 2010)*, LNCS 5977,

Springer, pp. 34–36, 2010. doi:10.1007/978-3-642-11811-1_4

[87] J. Hassine, “Early Schedulability Analysis with Timed Use Case Maps”, in *SDL 2009: Design for Motes and Mobiles, 14th Int. SDL Forum*, LNCS 5719, Springer, pp. 98–114, Sept. 2009. doi:10.1007/978-3-642-04554-7_7

[88] J. Hassine, J. Rilling, and R. Dssouli, “An ASM Operational Semantics for Use Case Maps”, in *13th IEEE Int. Requirement Engineering Conf. (RE05)*, IEEE CS, pp. 467–468, September 2005. doi:10.1109/RE.2005.10

[89] J. Hassine, J. Rilling, and R. Dssouli, “Timed Use Case Maps”, in *SAM 2006: Language Profiles - Fifth Workshop on System Analysis and Modelling*, LNCS 4320, Springer, pp. 99–114, 2006. doi:10.1007/11951148_7

[90] J. Hassine, J. Rilling and R. Dssouli, “Formal Verification of Use Case Maps with Real Time Extensions”, in *SDL-Forum 2007*, LNCS 4745, Springer, pp. 225–241, 2007. doi:10.1007/978-3-540-74984-4_14

[91] J. Hassine, J. Rilling, and R. Dssouli, “Use Case Maps as a property specification language”, *Software and Systems Modeling*, 8(2), pp. 205–220, 2009. doi:10.1007/s10270-007-0076-6

[92] J. Hassine, J. Rilling, and R. Dssouli, “An evaluation of timed scenario notations”. *Journal of Systems and Software*, 83(2), pp. 326–350, 2010. doi:10.1016/j.jss.2009.09.014

[93] Y. He, D. Amyot, and A. Williams, “Synthesizing SDL from Use Case Maps: An Experiment”, in *11th SDL Forum (SDL'03)*, LNCS 2708, Springer, pp. 117–136, July 2003. doi:10.1007/3-540-45075-0_7

[94] J. Hewitt and J. Rilling, “A Light-Weight Proactive Software Change Impact Analysis Using Use Case Maps”, in *IEEE Int. Workshop on Software Evolvability*, IEEE CS, pp. 41–46, 2005. doi:10.1109/IWSE.2005.1

[95] P. Heymans, G. Saval, G. Dallons, and I. Pollet, “Chapter VIII: A Template-Based Analysis of GRL”. *Advanced Topics in Database Research*, IGI Publishing, pp. 124–147, 2006. doi:10.4018/978-1-59140-935-9.ch008

[96] J. Hodges and J. Visser, “Accelerating Wireless Intelligent Network Standards Through Formal Techniques”, in *IEEE 1999 Vehicular Technology Conference*, IEEE CS, pp. 737–742, 1999. doi:10.1109/VETEC.1999.778276

[97] *i* Wiki*, http://istar.rwth-aachen.de/ (last accessed: July 12, 2010).

[98] IBM, *Rational DOORS*, USA, November 2010. http://www.ibm.com/software/awdtools/doors/

[99] ISO, Information Processing Systems, Open Systems Interconnection, *LOTOS — A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour*. IS 8807, 1989.

[100] ITU-T, Recommendation I.130 (11/88), *Method for the characterization of telecommunication services supported by an ISDN and network capabilities of an ISDN*. November 1988.

[101] ITU-T, Recommendation M.3020 (07/07), *Management interface specification methodology*. July 2007.

[102] ITU-T, Recommendation Z.100 (11/07), *Specification and Description Language*. November 2007.

[103] ITU-T, Recommendation Z.111 (11/08), *Notations to Define ITU-T Languages*. November 2008.

[104] ITU-T, Recommendation Z.120 (04/04), *Message Sequence Chart (MSC)*. April 2004.

[105] ITU-T, Recommendation Z.150 (02/03), *User Requirements Notation (URN) – Language Requirements and Framework*. February 2003.

[106] ITU-T, Recommendation Z.151 (11/08), *User Requirements Notation (URN) – Language definition*. November 2008. http://www.itu.int/rec/T-REC-Z.151/en

[107] M. Jackson, *Problem Frames: Analyzing and Structuring Software Development Problem*. Addison-Wesley, 2001.

[108] S. Jaskó, T. Dulaia, D. Muhia, and K. Tarnaya, “Test aspect of requirement specification”. *Computer Standards & Interfaces*, 32(1-2), pp. 1–9, January 2010. doi:10.1016/j.csi.2008.12.005

[109] B. Jiang, *Combining Graphical Scenarios with a Requirements Management System*. M.Sc. thesis, SITE, University of Ottawa, Canada, June 2005.

[110] *jUCMNav 4.3*, University of Ottawa, Canada, September 2010. http://jucmnav.softwareengineering.ca/jucmnav

[111] C. Kaewkasi and W. Rivepiboon, “WWM: a practical methodology for Web application modeling”, in *26th Annual Int. Computer Software and Applications Conf. (COMPSAC 2002)*, IEEE CS, pp. 603–608, August 2002. doi:10.1109/CMPSAC.2002.1045070

[112] P. Karpati, G. Sindre and A.L. Opdahl, “Visualizing Cyber Attacks with Misuse Case Maps”, in *16th Int. Working Conf. on Requirements Engineering: Foundation for Software Quality (REFSQ 2010)*, LNCS 6182, Springer, pp. 262–275, June 2010. doi:10.1007/978-3-642-14192-8_24

[113] R. Kazman and S.J. Carrière, “Playing Detective: Reconstructing Software Architecture from Available Evidence”. *Automated Software Engineering*, 6(2), pp. 107–138, 1999. doi:10.1023/A:1008781513258

[114] J. Kealey, *Enhanced Use Case Map Analysis and Transformation Tooling*. M.Sc. thesis, SITE, University of Ottawa, Canada, September 2007.

[115] J. Kealey and D. Amyot, “Enhanced Use Case Map Traversal Semantics”, in *13th SDL Forum (SDL'07)*, LNCS 4745, Springer, pp. 133–149, September 2007. doi:10.1007/978-3-540-74984-4_9

[116] J. Kealey, Y. Kim, D. Amyot, and G. Mussbacher, “Integrating an Eclipse-Based Scenario Modeling Environment with a Requirements Management System”, in *2006 IEEE Canadian Conf. on Electrical and Computer Engineering* (CCECE06), IEEE CS, pp. 2432–2435, May 2006.

[117] J. Kealey, E. Tremblay, J.-P. Daigle, J. McManus, O. Clift-Noël, and D. Amyot, “jUCMNav: une nouvelle plateforme ouverte pour l'édition et l'analyse de modèles UCM”, in *5ième Nouvelles Technologies de la Répartition (NOTERE 2005)*, Gatineau, Canada, pp. 215–222, August 2005.

[118] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, “Systematic literature reviews in software engineering - A systematic literature review”. *Inf. Softw. Technol.* 51, 1, pp. 7–15, Jan. 2009. doi:10.1016/j.infsof.2008.09.009

[119] C. Kuziemsky, X. Liu, and L. Peyton, “Leveraging Goal Models and Performance Indicators to Assess Health Care Information Systems”. 7th *Int. Conf. on the Quality of Information and Communications Technology (QUATIC 2010)*, IEEE CS, Porto, Portugal, September 2010. doi:10.1109/QUATIC.2010.37

[120] E. Lavendelis and J. Grundspenkis, “MASITS - A Tool for Multi-Agent Based Intelligent Tutoring System Development”, in *7th Int. Conf. on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2009)*, Advances in Soft. Computing Vol. 55, Springer, pp. 490–500, March 2009. doi:10.1007/978-3-642-00487-2_52

[121] P. Leelaprute, M. Nakamura, K. Matsumoto, and T. Kikuno, “Design and Evaluation of Feature Interaction

Filtering with Use Case Maps". *NECTEC Technical Journal*, 5(16) pp. 581–597, December 2005.

[122] T. Lethbridge and J. Singer, "Studies of the Work Practices of Software Engineers". *Advances in Software Engineering: Comprehension, Evaluation and Evolution*, Springer-Verlag, pp. 51–72, 2002.

[123] H. Liu, *Multilevel Performance Analysis of Scenario Specification for a Presence System*. M.Sc. thesis, SCE Dept., Carleton University, Canada, October 2002.

[124] L. Liu and E. Yu, "From requirements to architectural design—using goals and scenarios", in *ICSE-2001 Workshop: From Software Requirements to Architectures (STRAW 2001)*, Toronto, Canada, pp.22–30, May 2001.

[125] L. Liu and E. Yu, "Designing Information Systems in Social Context: A Goal and Scenario Modelling Approach". *Information Systems*, pp. 187–203, April 2004. doi:10.1016/S0306-4379(03)00052-8

[126] L. Liu and E. Yu, *GRL - Goal-oriented Requirement Language*, 2000. http://www.cs.toronto.edu/km/GRL/

[127] R. Matulevičius, P. Heymans, and A. L. Opdahl, "Comparing GRL and KAOS using the UEML Approach". *Enterprise Interoperability II: New Challenges and Approaches*, August 2007, pp. 77-88

[128] A. Medve, "Advanced steps with standardized languages in the re-engineering process". *Computer Standards & Interfaces*, 30(5), Elsevier, p. 315–322, July 2008. doi:10.1016/j.csi.2007.09.004

[129] A. Miga, *Application of Use Case Maps to System Design With Tool Support*. Master's thesis, SCE Dept., Carleton University, Canada, October 1998.

[130] A. Miga, D. Amyot, F. Bordeleau, D. Cameron, and M. Woodside, "Deriving Message Sequence Charts from Use Case Maps Scenario Specifications", in *Meeting UML - Tenth SDL Forum (SDL'01)*, LNCS 2078, Springer, pp. 268–287, June 2001. doi:10.1007/3-540-48213-X_17

[131] F. Molina, J. Pardillo, C. Cachero, and A. Toval, "An MDE Modeling Framework for Measurable Goal-Oriented Requirements". *Int. J. of Intelligent Systems*, 25(8), Wiley, pp. 757–783, August 2010. doi:10.1002/int.20430

[132] O. Monkewich, I. Sales, and R. L. Probert, "OSPF Efficient LSA Refreshment Function in SDL", in *Tenth SDL Forum (SDL'01)*, LNCS 2078, Springer, June 2001, pp. 300–315. doi:10.1007/3-540-48213-X_19

[133] D.L. Moody, P. Heymans, and R. Matulevičius, "Visual syntax does matter: improving the cognitive effectiveness of the i* visual notation". *Requirements Engineering*, 15(2), Springer, pp.141–175, 2010. doi:10.1007/s00766-010-0100-1

[134] G. Mussbacher, *Aspect-oriented User Requirements Notation*. Ph.D. thesis, SITE, University of Ottawa, Canada, November 2010.

[135] G. Mussbacher and D. Amyot, "A Collection of Patterns for Use Case Maps", in *First Latin American Conf. on Pattern Languages of Programming (SugarLoafPLoP)*, UERJ - Série Informática, Special Edition, pp. 57–82, June 2002.

[136] G. Mussbacher and D. Amyot, "Assessing the Applicability of Use Case Maps for Business Process and Workflow Description", in *3rd Int. MCeTech Conference on eTechnologies*, IEEE CS, pp. 219–222, January 2008. doi:10.1109/MCETECH.2008.18

[137] G. Mussbacher and D. Amyot, "Goal and Scenario Modeling, Analysis, and Transformation with jUCMNav", in *31st Int. Conf. on Software Engineering (ICSE-Companion)*, ACM, Canada, pp. 431–432, May 2009. doi:10.1109/ICSE-COMPANION.2009.5071047

[138] G. Mussbacher, D. Amyot, J. Araújo, and A. Moreira, "Modeling Software Product Lines with AoURN", in *Early Aspects Workshop @ AOSD08*, ACM, March 2008. doi:10.1145/1404946.1404948

[139] G. Mussbacher, D. Amyot, J. Araújo, and A. Moreira, "Requirements Modeling with the Aspect-oriented User Requirements Notation (AoURN): A Case Study". *Transactions on Aspect-Oriented Software Development VII*, LNCS 6210, Springer, pp. 23–68, 2010. doi:10.1007/978-3-642-16086-8_2

[140] G. Mussbacher, D. Amyot, and M. Weiss, "Formalizing Patterns with the User Requirements Notation". *Design patterns formalization techniques*, IGI Global, pp. 302–322, 2007. doi:10.4018/978-1-59904-219-0.ch014

[141] G. Mussbacher, D. Amyot, and M. Weiss, "Visualizing Early Aspects with Use Case Maps". *LNCS Journal on Transactions on Aspect-Oriented Software Development*, LNCS 4620, Springer, p. 105–143, November 2007. doi:10.1007/978-3-540-75162-5_5

[142] G. Mussbacher, D. Amyot, and J. Whittle, "Refactoring-Safe Modeling of Aspect-Oriented Scenarios", in *12th Int. Conf. on Model Driven Eng. Languages and Systems (MODELS 2009)*, LNCS 5795, Springer, pp. 286–300, October 2009. doi:10.1007/978-3-642-04425-0_21

[143] G. Mussbacher, J. Whittle, and D. Amyot, "Modeling and Detecting Semantic-Based Interactions in Aspect-Oriented Scenarios". *Requirements Engineering*, 15(2), Springer, pp.197-214, 2010. doi:10.1007/s00766-010-0098-4

[144] M. Nakamura, T. Kikuno, J. Hassine, and L. Logrippo, "Feature Interaction Filtering with Use Case Maps at Requirements Stage", in *Sixth International Workshop on Feature Interactions in Telecommunications and Software Systems (FIW'00)*, IOS Press, pp. 163–178, May 2000.

[145] Object Management Group, *BPMN 1.2 Specification*, formal/2009-01-03, January 2009.

[146] Object Management Group, *UML 2.2 Specification*, formal/2009-02-04, February 2009.

[147] Object Management Group, *UML Profile for Schedulability, Performance and Tim*e, v1.0, formal/03-09-01, September 2003.

[148] Object Management Group, *UML Profile for Modeling and Analysis of Real-time and Embedded Systems (MARTE)*, v1.0, formal 2009-11-02, November 2009.

[149] C. Ölvingson, N. Hallberg, T. Timpka, and K. Lindqvist, "Requirements Engineering for Inter-Organizational Health Information Systems with Functions for Spatial Analyses: Modeling a WHO Safe Community Applying Use Case Maps". *Methods of Information in Medicine*, Schattauer Gmb H, 4/2002, pp. 299–304, 2002.

[150] OpenOME, an open-source requirements engineering tool, University of Toronto, Canada, November 2010. https://se.cs.toronto.edu/trac/ome

[151] T. Okamura, M. Nakamura, and H. Igaki, "Finding Optimal Energy-Saving Operations in Home Network System Based on Effects between Appliances and Environment", in 8th *Asia-Pacific Symp. on Information and Telecommunication Technologies (APSITT 2010)*, IEEE CS, Kuching, Malaysia, , pp. 1–6, June 2010.

[152] D.B. Petriu, *Layered Software Performance Models Constructed from Use Case Map Specifications*. M.Eng. thesis, SCE Dept., Carleton University, Canada, May 2001.

[153] D.B. Petriu, D. Amyot, and C.M. Woodside, "Scenario-Based Performance Engineering with UCMNav", in *11th SDL Forum (SDL'03)*, LNCS 2708, Springer, pp. 18–35, July 2003. doi:10.1007/3-540-45075-0

[154] D.B. Petriu and C.M. Woodside, "Software performance models from system scenarios". *Performance Evaluation*, 61(1), Elsevier, pp. 65–89, June 2005. doi:10.1016/ j.peva.2004.09.005

[155] D.B. Petriu and C.M. Woodside, "An intermediate metamodel with scenarios and resources for generating performance models from UML designs". *Software and Systems Modeling*, 6(2), Springer, pp. 163–184, June 2007. doi:10.1007/s10270-006-0026-8

[156] A. Pourshahid, *A URN-Based Methodology for Business Process Monitoring*, M.Sc. thesis, EBT, University of Ottawa, Canada, March 2008.

[157] A. Pourshahid, P. Chen, D. Amyot, A.J. Forster, S. Ghanavati, L. Peyton, and M. Weiss, "Business Process Management with the User Requirements Notation". *Electronic Commerce Research*, 9(4), Springer, pp. 269–316, December 2009. doi:10.1007/s10660-009-9039-z

[158] A. Pourshahid, G. Mussbacher, D. Amyot, and M. Weiss, "Toward an Aspect-Oriented Framework for Business Process Improvement". *Int. J. of Electronic Business*, 8(3), Inderscience Publisers, pp. 233–259, 2010. doi:10.1504/IJEB.2010.034170

[159] A. Pourshahid and T. Tran, "Toward an Effective Trust Management System for E-Commerce: Modeling Trust Components and Processes Using URN". *Journal of Business and Technology (JBT)*, 2(2), Atlantic Academic Press, pp. 37–46, 2007.

[160] J.-F. Roy, J. Kealey, and D. Amyot, "Towards Integrated Tool Support for the User Requirements Notation", in *SAM 2006: Language Profiles - Fifth Workshop on System Analysis and Modelling*, LNCS 4320, Springer, pp. 198–215, May 2006. doi:10.1007/11951148_13

[161] J.-F. Roy, *Requirement Engineering with URN: Integrating Goals and Scenarios*. M.Sc. thesis, SITE, University of Ottawa, Canada, March 2007

[162] N. Russell, A.H.M. ter Hofstede, W.M.P. van der Aalst, and N. Mulyar, *Workflow Control-Flow Patterns: A Revised View*. BPM Center Report BPM-06-22, 2006. http://workflowpatterns.com/

[163] Sandrila Ltd., *Sandrila SDL*. UK, November 2010. http://www.sandrila.co.uk

[164] H. Saiedian, P. Kumarakulasingam, and M. Anan, "Scenario-Based Requirements Analysis Techniques for Real-Time Software Systems: A Comparative Evaluation". *Requirements Engineering*, 10(1), Springer, pp. 22–33, January 2005. doi:10.1007/s00766-004-0192-6

[165] K. Saleh and A. Al-Zarouni, "Capturing Non-Functional Software Requirements using the User Requirements Notation", in *2004 Int. Research Conf. on Innovation in Information Technology (IIT'04)*, Dubai, pp. 222–230, October 2004.

[166] K. Saleh and G. Elshahry, "Modeling Security Requirements for Trustworthy Systems". *Encyclopedia of Information Science and Technology*, 2nd edition, IGI Global, pp. 2657–2664, 2009. doi:10.4018/978-1-60566-026-4.ch424

[167] I. Sales, *A Bridging Methodology for Internet Protocols Standards Development*. M.Sc. thesis, SITE, University of Ottawa, Canada, August 2001.

[168] I.S. Sales and R.L. Probert, "From High-Level Behaviour to High-Level Design: Use Case Maps to Specification and Description Language", in *18th Brazilian Symp. on Computer Networks (SBRC2000)*, Brazil, May 2000.

[169] W.C. Scratchley, *Evaluation and Diagnosis of Concurrency Architectures*. Ph.D. thesis, SCE Dept., Carleton University, Canada, July 2000.

[170] W.C. Scratchley and C.M. Woodside, "Evaluating Concurrency Options in Software Specifications", in *Seventh Int. Symp. on Modelling, Analysis and Simulation of Computer and Telecom. Systems (MASCOTS'99)*, College Park, USA, pp. 330–338, October 1999. doi:10.1109/MASCOT.1999.805071

[171] S. Schneider, *The B-Method: An Introduction*, Palgrave, Cornerstones of Computing series, 2001.

[172] A. Shamsaei, A. Pourshahid, and D. Amyot, "Business Process Compliance Tracking Using Key Performance Indicators", in *6th Int. Workshop on Business Process Design (BPD 2010)*, LNBIP 66, Springer, pp. 73–84, September 2010.

[173] M. Shiri, *Supporting UCM Requirements Evolution by Means of Formal Concept Analysis*. M.Sc. thesis, CSCE dept., Concordia University, Canada, February 2008.

[174] M. Shiri, J. Hassine, and J. Rilling, "Feature Interaction Analysis: A Maintenance Perspective", in *22nd IEEE/ACM Int. Conf. on Automated Software Engineering (ASE)*, ACM Press, pp. 437–440, November 2007. doi:10.1145/1321631.1321703

[175] K.H. Siddiqui and C. M. Woodside, "Performance Aware Software Development (PASD) Using Resource Demand Budgets", in *WOSP 2002: Third Int. Work. on Software and Performance*, ACM Press, pp. 275-285, July 2002. doi:10.1145/584369.584412

[176] G. Smith, *The Object-Z Specification Language*. Advances in Formal Methods Series, Kluwer Academic Publishers, 2000.

[177] Software Performance Research Group, *Layered Queueing Research Resource Page*. Carleton University, Canada, November 2010. http://www.layeredqueues.org/

[178] A. Soria, J.A. Díaz-Pace, and M.R. Campo, "Tool Support for Fault Localization Using Architectural Models", *13th European Conf. on Software Maintenance and Reengineering (CSMR)*, IEEE CS, March 2009, pp. 59–68. doi:10.1109/CSMR.2009.42

[179] C.H. Störmer, *Software Quality Attribute Analysis by Architecture Reconstruction (SQUA3RE)*, Ph.D. thesis, Vrije Universiteit, The Netherlands, March 2007.

[180] N.-T. Truong, T.M.T. Tran, V.-K. To, and V.H. Nguyen, "Checking the Consistency between UCM and PSM Using a Graph-Based Method", in *1st Asian Conf. on Intelligent Information and Database System (ACIIDS 09)*, IEEE CS, pp. 190–195, April 2009. doi:10.1109/ACIIDS.2009.66

[181] *Use Case Map Navigator (UCMNAV) 2.3*, July 2005. http://jucmnav.softwareengineering.ca/ucm/bin/view/UC M/UcmNav

[182] *URN Virtual Library*, http://www.UseCaseMaps.org/pub (last accessed: July 27, 2010).

[183] J.A. van der Poll, P. Kotze, A. Seffah, T. Radhakrishnan, and E. Alsumait, "Combining UCMs and Formal Methods for Representing and Checking the Validity of Scenarios as User Requirements", in *2003 annual research conf. of the South African Institute of Computer Scientists and Information Technologists (SAICSIT 2003)*, Johannesburg, South Africa, pp. 59–68, September 2003.

[184] B. van der Raadt, *Business-Oriented Exploration of Web Services Ideas - Combining Goal-Oriented and Value-Based Approaches*. MSc. Thesis, Vrije Universiteit, Amsterdam, The Netherlands, February 2005.

[185] A. van Lamsweerde, *Requirements engineering: From System Goals to UML Models to Software Specifications*. John Wiley & Sons, 2009.

[186] M. Vigder, *Applying Formal Techniques to the Design of Concurrent Systems*. Ph.D. thesis, SCE Dept., Carleton University, Canada, July 1992.
[187] M. Vinje, *An Auditing Framework for Service Provision in Mobile IPv6 Networks*. Diploma Thesis, EPF Zurich, Switzerland, August 2002.
[188] M. Weiss, "Pattern-Driven Design of Agent Systems: Approach and Case Study", in *15th Conf. on Advanced Information Systems Engineering (CAiSE'03)*, LNCS 2681, Springer, pp. 711–723, June 2003. doi:10.1007/3-540-45017-3
[189] M. Weiss, "More Patterns for Web Applications", in *Tenth European Conf. on Pattern Languages of Programs (Euro PLoP 2005)*, Irsee, Germany, pp. 21–34, July 2005.
[190] M. Weiss and D. Amyot, "Business process modeling with URN," *Int. J. of E-Business Research*, 1(3), pp. 63–90, 2005. doi:10.4018/jebr.2005070104
[191] M. Weiss and B. Esfandiari, "On Feature Interactions among Web Services". *Int. J. of Web Services Research*, 2(4) pp. 22–47, October 2005. doi:10.4018/jwsr.2005100102
[192] M. Weiss, B. Esfandiari, and Y. Luo, "Towards a classification of Web service feature interactions". *Computer Networks*, 51(2), Elsevier, pp. 359–381, February 2007. doi:10.1016/j.comnet.2006.08.003
[193] M. Weiss and H. Mouratidis, "Selecting Security Patterns that Fulfill Security Requirements", in *16th IEEE Int. Requirements Engineering Conf. (RE'08)*, IEEE CS, pp. 169–172, September 2008. doi:10.1109/RE.2008.32
[194] P. Wu and C.M. Woodside, "An Aggregation Approach to Constructing Hybrid Layered Queueing Models", in *7th Int. Workshop on Performability Modeling of Computer and Communication Systems (PMCCS7)*, Torino, Italy, September 2005.
[195] W. Wu and T.P. Kelly, "Managing Architectural Design Decisions for Safety-Critical Software Systems", in *2nd Int. Conf. on the Quality of Software Architectures (QoSA 2006)*, LNCS 4126, Springer, pp. 59–77, June 2006. doi:10.1007/11921998_9
[196] Z. Yi, CNAP Specification and Validation: A Design Methodology Using LOTOS and UCM. M.Sc. thesis, SITE, University of Ottawa, Canada, January 2000.
[197] E.S.-K. Yu, *Modelling strategic relationships for process reengineering*. Ph.D. thesis, Dept. of Computer Science, University of Toronto, Canada, 1995.
[198] E. Yu, Y. Yu, and L. Liu, *OME — Organization Modelling Environment*, University of Toronto, 2000. http://www.cs.toronto.edu/km/ome/
[199] X.Y. Zeng, *Transforming Use Case Maps to the Core Scenario Model Representation*. M.Sc. thesis, SITE, University of Ottawa, Canada, June 2005
[200] R. Zhang and X.-X. Liu, "Feature Interaction Filtering Method Based on URN (基于 URN 的特征冲突过滤方法)", *Computer Engineering (计算机工程)*, 35(21), pp. 45–47, November 2009.

Daniel Amyot received both his Ph.D. (2001) and M.Sc. (1994) degrees in computer science from the University of Ottawa. The research topic was related to the specification and validation of telecommunication systems with Use Case Maps and LOTOS.

After working for Mitel Networks as a senior researcher, he joined the School of Information Technology and Engineering of the University of Ottawa, where he is now Associate Professor in software engineering. His research interests include goal-oriented and scenario-based software engineering, requirements engineering, business process modeling, aspect-oriented modeling, and healthcare informatics. He has published over 90 papers in various conferences and in journals such as Requirements Engineering, Computer Networks, and the International Journal of Electronic Business.

Dr. Amyot is a member of ACM, IEEE Computer Society, and APIIQ, and he is a professional engineer in the province of Québec (Canada). He is also Associate Rapporteur for requirements languages at the International Telecommunication Union, where he leads the evolution of the User Requirements Notation.

Gunter Mussbacher received a M.Sc. degree in computer science from Simon Fraser University in 1999, and a Ph.D. in computer science from the University of Ottawa in 2010. In his thesis, he developed the Aspect-oriented User Requirements Notation (AoURN), a framework that enables goal-oriented, scenario-based, and aspect-oriented modeling in a unified way.

After his M.Sc., he worked as a research engineer for the Strategic Technology department of Mitel Networks, where he applied and taught URN concepts. He has published in the Requirements Engineering Journal (REJ) and in the Transactions on Aspect-Oriented Software Development (TAOSD), and co-edited with Daniel Amyot the URN standard (ITU Recommendation Z.151 11/2008). He is also teaching software engineering undergraduate courses as well as URN and AoURN tutorials for industry and at international conferences. His general research interests lie in requirements engineering, URN, aspect-oriented modeling, and patterns.

Dr. Mussbacher is an organizer and program committee member of Early Aspects (EA), Aspect-oriented Modeling (AOM), and Systems Analysis and Modelling (SAM) workshops since 2008.