# Combining Goal modelling with Business Process modelling

## Two Decades of Experience with the User Requirements Notation Standard

Daniel Amyot[*,a,b], Okhaide Akhigbe[a], Malak Baslyman[c], Sepideh Ghanavati[d], Mahdi Ghasemi[a], Jameleddine Hassine[c], Lysanne Lessard[a,b], Gunter Mussbacher[e], Kai Shen[a], Eric Yu[f]

[a] University of Ottawa, Ottawa, Canada.
[b] Institut du savoir Montfort, Ottawa, Canada.
[c] King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia.
[d] University of Maine, Orono, USA.
[e] McGill University, Montreal, Canada
[f] University of Toronto, Toronto, Canada

Abstract. *Goal modelling aims to capture stakeholder and system goals, together with social, intentional, and structural relationships, in a way that supports trade-off analysis and decision making. Goal models and business process models provide complementary and synergetic views of a system, which lead to a more complete understanding of what exists and a better description of what needs to be designed than with only one of these views. The User Requirements Notation (URN), standardized in 2008 by the International Telecommunication Union (ITU-T) with improved versions in 2012 and 2018, combines goal modelling with process modelling, while providing graphical and textual syntaxes for both views. URN helps modellers exploit the value of social modelling in the context of process design and improvement. In this paper, we report on nearly two decades of combined goal/process modelling with URN in different areas – based on coarse-grained statistics from a literature review and the authors' personal experiences with URN. In particular, beyond the telecommunication services for which URN was created, we highlight applications to goal/process alignments, regulatory compliance and intelligence, process adaptation and improvement, value co-creation and service systems, and goal-oriented process mining, as well as several advanced modelling techniques. An overview of the tool-supported analyses (for satisfaction, alignment, compliance, and others) and model creation mechanisms (composition, aspects, slicing, adaptation, reuse, and others) is also provided. The last part of this paper focuses on important challenges and exciting opportunities for future research, especially in the areas of data-driven applications (e.g, AI/machine learning), socio-cyber-physical systems, and usable automation.*

Keywords. Goal-oriented modelling • GRL • Process modelling • UCM • User Requirements Notation

Communicated by Ralf Laue. Received 2021-03-15. Accepted after 1 revision on 2021-11-05.

## 1 Introduction

Process modelling languages have now been around for a century, and the seminal work on Pro-

* Corresponding author.
E-mail. damyot@uottawa.ca

This paper is dedicated to Raymond Buhr, inventor of the Use Case Maps notation, who passed away in January 2021.

We thank the many people involved in the development of URN and of related methods, standards, and tools since its inception, as well as the anonymous referees for their useful feedback. This work was supported in part by NSERC.

cess Charts by Gilbreth and Gilbreth (1921) has since influenced dozens of variants and successors (Mili et al. 2010). These domain-specific modelling languages (Karagiannis et al. 2016) enable the representation and analysis of the activities of a process (*what*), their sequencing (*when*), their sub-activities (*how*), and their performers (*who/where*). While contemporary process modelling languages such as the Business Process Model and Notation (BPMN) (OMG 2014), Activity Diagrams (OMG 2017), Petri Nets (ISO 2019), and Integration Definition 0 (IDEF0) (IEEE 2012) can help engineers and business analysts reason about what, when, where, who, and how questions, they are rather limited in addressing *why* questions that are important for trade-off analysis and for rationale documentation, among others.

Goal modelling languages, on the other hand, excel at describing rationales of systems and processes, together with social aspects and dependencies. These languages typically describe goals (*what/why*), their refinements into activities (*how*), and various relationships to performers (*who* and *for whom*) and to each others (*why*). They thus provide important information about the *who*'s and *why*'s of a system or organizational context, but they do not express sequencing, which is one other major difference they have with process modelling languages.

Over the past 25 years, different goal modelling languages have been proposed (Horkoff et al. 2019), including *i** (Dalpiaz et al. 2016; Yu 1997; E. S. Yu et al. 2011), the Non-Functional Requirement Framework (Chung et al. 2000), GBRAM (Antón 1996), KAOS (van Lamsweerde 2001, 2009), the Business Intelligence Model (BIM) (Horkoff et al. 2014), and TROPOS (Bresciani et al. 2004). Some of these languages also have their own extensions and variants. For example, Gonçalves et al. (2018) have reviewed 96 papers presenting extensions of the *i** language. Many of these languages have industrial applications as well (Horkoff et al. 2019; E. Yu et al. 2013).

## 1.1 Process/Goal Combinations

Given that process modelling languages and goal modelling languages have complementary strengths regarding their modelling concepts and the types of analysis questions their models can answer, it becomes important to explore whether combining their capabilities can be beneficial. Over the years, several combinations have been explored, including this non-exhaustive list:

- Three decades ago, Godart et al. (1992) explored early ideas about combining goal modelling with an informal process modelling notation for specifying cooperative software development activities. Goals were described as logical predicates with positive and negative dependencies on activities, and a reasoning engine enabled simple planning.
- Jacobs and Holten (1995) provided one of the first integrations of simple process models with goal models based on a common metamodel implemented using TELOS (Mylopoulos et al. 1990). This combination enabled goals to drive decisions among explicit alternative activities at the process level. Contributions between goals were based on The House of Quality (Clausing and Hauser 1988), with support for fuzzy qualitative reasoning.
- Decreus et al. (2009) published a literature review of practical challenges in transforming goal models (in *i** here) to business process models in general (including BPMN). Many of these challenges, related to concept mapping, structures, sequencing, and automation, are still current a decade later.
- Transformations in the opposite direction were also explored. For example, Odeh et al. (2018) investigated how to derive *i** models (manually) from a category of business process models that are characterized as goal-based and role-oriented. How to do this for general process models remains an issue.
- In a healthcare context, Cardoso et al. (2011) explored traceability and harmonization between TROPOS goal models and ARIS-based process

models in order to improve goal-process alignment.

- Such alignment was also investigated by Koliadis and Ghose (2006), who have proposed a method that combines KAOS goal models with BPMN process models, with alignment verification in changing, dynamic contexts.
- Gol Mohammadi (2019) loosely combined BPMN with *i** to model trustworthiness requirements in cyber-physical systems, with explicit concepts for trustworthiness goals, concerns, and threats.

These examples show that different combinations were explored to serve specific purposes or domains. However, most approaches faced many challenges related to the combination of languages and tools that were not meant to interoperate in the first place. This, in turn, limits their analysis capabilities and potential benefits.

### 1.2 User Requirements Notation

One particular combination of two languages, namely Use Case Maps (UCM) for process modelling and the Goal-oriented Requirement Language (GRL) for goal modelling, resulted in the *User Requirements Notation* (URN). URN is a standard from the International Telecommunication Union (ITU-T) that explicitly addresses goals and processes in graphical and textual ways, in one unified conceptual modelling language, with links between the two perspectives (ITU-T 2018). This standard also includes original features such as measurable indicators and quantitative/qualitative analysis for goal models, as well as executable scenario definitions and the support for dynamic refinements in process models.

In this paper, the research question of interest is "*What are the benefits of combining goal modelling with process modelling?*". We are answering this question by focusing on two decades of experience in using URN in multiple academic and industrial contexts – based on coarse-grained statistics from a literature review and the authors' personal experiences with URN. Furthermore, we discuss existing and new application areas for URN to give insight into these benefits. Although UCM and GRL (or their own ancestors) were used individually in the 1990's, their synergical combination only started to appear about two decades ago. URN enables a rich set of language features, analysis capabilities, and application areas that justify its selection for answering this research question.

This paper contributes a contemporary overview of the URN language, of its general usage, and of the benefits brought by its combination of goal modelling with process modelling along many lines of research. This knowledge is beneficial to researchers interested in the synergy between goal and process modelling in general (and not just with URN), and to practitioners looking for immediate or potential solutions related to the application domains discussed in this paper.

This paper is structured as follows. Sect. 2 first provides a historical perspective on the standardization of URN, together with an introduction to URN's syntax, semantics, and analysis capabilities, through an illustrative example. Sect. 3 situates URN by providing a trend analysis of various areas where this language has been applied over the last 25 years. Then, Sections 4 to 9 discuss in more detail the benefits of using integrated goal and process models in the domains of goal/process alignment, regulatory compliance and intelligence, process adaptation and improvement, value co-creation and service systems, and goal-oriented process mining, as well as other advanced URN-based modelling techniques, respectively. These sections present a significant body of literature reporting on empirical studies in varied application areas, reflecting the experiences of dozens of researchers and practitioners in applying and evaluating URN. The benefits of using URN in these domains, and integrated goal and process models more generally, are highlighted for each domain. Sect. 10 presents research challenges and opportunities for the next decade of URN-based modelling. The conclusion summarizes the paper.

## 2 URN Overview

After giving a historical perspective, this section highlights, with the help of an illustrative example, the graphical and textual syntaxes of the process and goal modelling parts of URN. Basic analysis capabilities based on the URN semantics are also presented.

### 2.1 URN History

The history of URN's building blocks goes back nearly three decades. The Use Case Maps notation was first proposed by Buhr and Casselman (1995) from Carleton University, with a revised version in a seminal paper by Buhr (1998). However, UCM was itself based on earlier work by Vigder (1992) on design *slices*, under Buhr's supervision. Slices helped visualize the behaviour of cross-cutting concurrent software components. They were extended to support the description of scenarios in object-oriented and real-time systems and became *timethreads*, which were themselves renamed Use Case Maps around 1995. UCM was then presented as a notation for "describing, in a high-level way, how the organizational structure of a complex system and the emergent behaviour of the system are intertwined" (Buhr 1998). The notation was not meant to be executable or analyzable through automated means.

Around the same period, several researchers at the University of Toronto developed two modelling notations that would form the basis of GRL, namely the Non-Functional Requirements (NFR) Framework (Chung et al. 2000; Mylopoulos et al. 1992), and the *i** framework (Yu 1997; E. S. Yu et al. 2011). The first version of GRL was produced by Liu and Yu around 2000[1] and proposed a graphical syntax based mainly on a subset of *i** that focused on intentional elements, links, and actors (for social modelling), supplemented with additional concepts supporting the kind of propagation-based analysis found in the NFR framework. A textual syntax and an interchange XML schema were also proposed.

[1] See https://www.cs.toronto.edu/km/GRL/

In 1999, a large telecommunication company (Nortel), heavily involved in international standardization activities, was looking for a new language that would enable engineering standardization bodies to describe new wireless telephony services in an abstract way. Their main problem was that the sequence diagrams and Message Sequence Charts used as notations at the time required a commitment to networking infrastructure that varied substantially across companies and countries. They became interested in UCMs as they decoupled behaviour from the underlying structure of components. UCMs were also used in other companies including Mitel, which was also in favor of standardization. Mitel was using goal modelling for documenting architectural decisions, and they suggested combining goal and scenario modelling as part of the proposal for a new standard, especially to support reflective systems (taking social aspects into consideration), and for feature personalization. Motorola also supported this way forward.

Requirements for the new language were first approved by ITU-T in 2003 as Recommendation Z.150, which was updated eight years later (ITU-T 2011). The URN language description itself was first standardized in 2008 as Recommendation Z.151 (ITU-T 2018), under the leadership of Amyot and Mussbacher from the University of Ottawa. The 2008 URN standard contains a definition of the abstract syntax with an integrated metamodel for goal and process modelling, a graphical syntax for UCM and GRL, an XML-based interchange format, and modelling elements supporting the definition of analysis contexts. URN was revised in 2012 mainly to support indicators in GRL, and to improve the execution mechanism of UCM models (called *traversal*). The 2018 version of the standard now includes a textual syntax for URN, similar to a programming language.

In the URN standard, the semantics of URN are specified with a set of detailed requirements for the execution of UCM models and a detailed description of GRL analysis algorithms. Furthermore, the execution semantics of UCM models have been formalized outside the standard with

the LOTOS process algebra (Amyot and Logrippo 2000) as well as with Abstract State Machines (Hassine et al. 2005a,b). GRL, on the other hand, has been formalized by mapping GRL models to constraint satisfaction problems (Fan et al. 2018). Other historical details are provided in a previous survey (Amyot and Mussbacher 2011).

Interestingly, the URN standard did not at first present UCM as a process modelling language, but rather as a *scenario* modelling language. This changed later once URN started being used heavily for business process management.

## 2.2 Graphical Syntax

URN's graphical syntax is introduced here through an illustrative example. An *Incident Management System* (IMS) is typically used in a hospital to report and categorize incidents, which may be clinical (e. g., incorrect medication dosage administered) or not (e. g., a visitor slides and falls on the floor). An incident is first reported by an observer (e. g., an employee or a visitor), and then reviewed by a committee to categorize it. This is important in order to mitigate such incidents in the future, both to improve patient safety and avoid potential legal actions. The review committee must include a physician in case the incident is of a clinical nature. As the committee needs to be efficient, the IMS can provide advanced features to automatically request and find available physicians. Another feature that can improve effectiveness is the early detection of suspected duplicate incidents, which can be quickly confirmed by the original observer.

### 2.2.1 UCM

Fig. 1 shows a UCM diagram (called a *map*) describing this process. Note that the path highlighted in red in the figure represents one specific scenario supported by the process model; this will be explained further in Sect. 2.4.

A UCM map allows for the process elements (start/end points, responsibilities, forks/joins, stubs, timers, and others presented later in this section) to be allocated to *components*, which can be of different natures. A component may be of type *team* ▭ (e. g., IMS) and represent the system itself or a part of the system at any level of granularity.

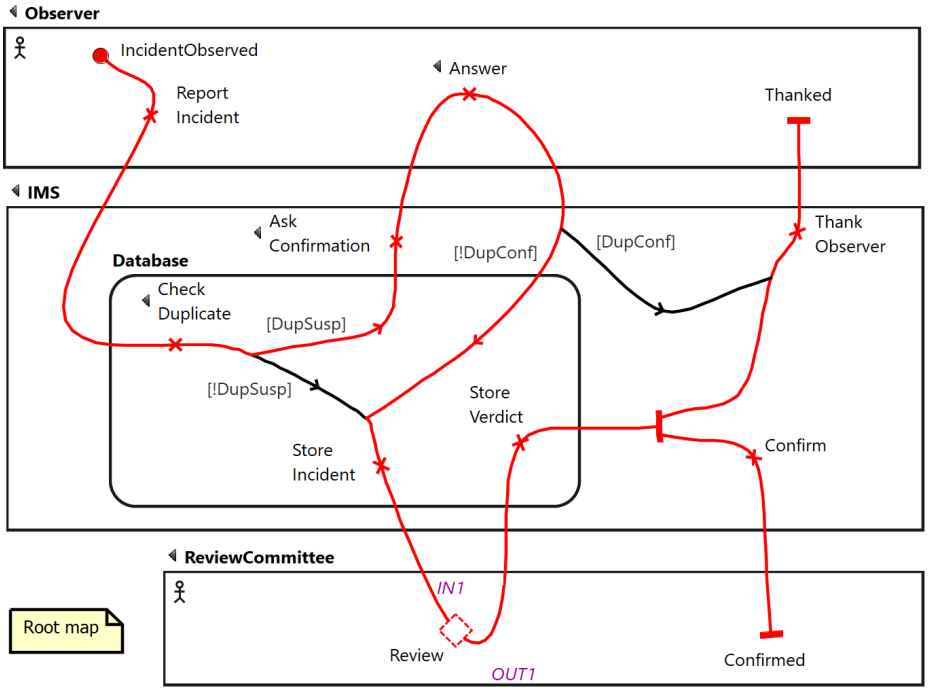
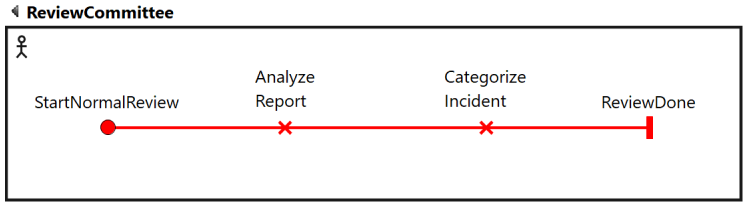
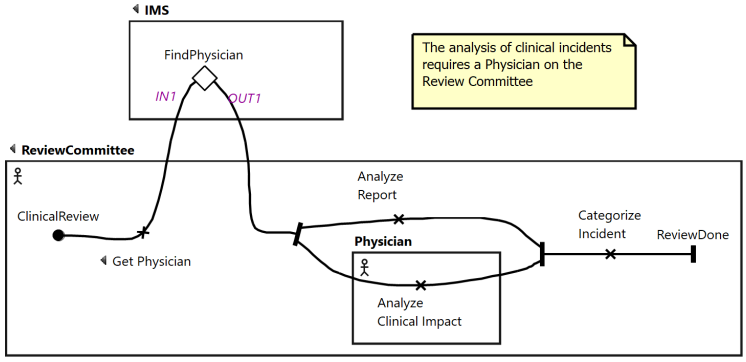


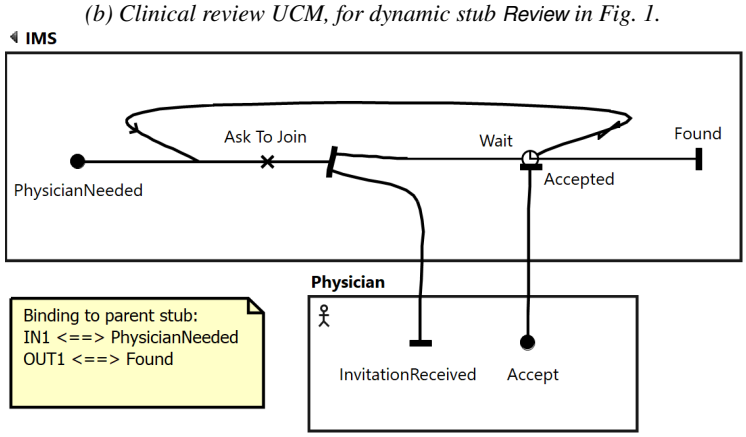

*Figure 1: Incident reporting and categorization process in an IMS: Root map in UCM.*




*(a) Normal review UCM, for dynamic stub Review in Fig. 1.*




*(b) Clinical review UCM, for dynamic stub Review in Fig. 1.*




*(c) Physician finding UCM, for stub FindPhysician in Fig. 2b.*

*Figure 2: Sub-processes, also modelled as UCM maps, bound to their containing stubs.*

A component of type *actor* (e. g., Observer and ReviewCommittee) describe anyone or anything that interacts with the system. Components can also contain other components, and a system component may be deemed a passive component of type *object* (e. g., the IMS contains a passive component representing the incident Database).

A map has at least one *start point* ● expressing a precondition and/or triggering event (e. g., IncidentObserved), and at least one *end point* ▮ expressing a postcondition and/or resulting event (e. g., Thanked and Confirmed). *Responsibilities* × (e. g., Report Incident) describe the activities performed along a path. Alternatives are captured with OR-forks and their two or more forked paths can be guarded with conditions (e. g., [!DubSusp], where ! represents negation and DubSusp is a Boolean variable that is true iff a duplicate incident is suspected). Paths can also merge with an OR-join, but well-nestedness is not required by the language. Similarly, concurrency can be introduced along AND-forks and paths can be synchronized via AND-joins.

As in many other modelling languages, complex

processes can be decomposed into reusable sub-processes. A UCM *stub* ◇ is a container for one sub-process, often called *plug-in* map in the UCM literature. Whereas a normal stub contains only one plug-in, a *dynamic stub* ◇ contains two or more plug-ins whose selection is determined at run-time according to a selection policy composed of a guarding condition for each contained plug-in. For example, the Review dynamic stub in Fig. 1 contains two plug-in UCMs, one for the normal review process in Fig. 2a (guarded by the condition !Clinical) and one for the clinical review process in Fig. 2b (guarded by Clinical).

There could be many levels of decomposition. For example, the stub FindPhysician in Fig. 2b contains a third level of process decomposition with a plug-in map (Fig. 2c) that describes how the IMS finds a physician to participate in the review committee. This particular process uses a UCM *timer* ◷ that is reset if the physician accepts the invitation in a timely way, and that times-out and loops back otherwise.

There are two important characteristics of UCM modelling illustrated here. First, new components and containment relationships can be introduced in sub-processes. Second, some start points and end points act as connectors to their parent stub, to ensure continuity of processes across decomposition levels. The binding between the map in Fig. 2c and its parent stub (FindPhysician) indicates which input segment of the stub is connected to which start point (IN1 with PhysicianNeeded here) and which output of the stub is connected to which end point (OUT1 with Found here). Unbound start/end points describe local triggering/resulting events.

### 2.2.2 GRL

From a goal modelling perspective, Fig. 3 presents the GRL view of the IMS and its stakeholders, all represented with *actors* ⬭. Actors have intentional elements such as *goals* ⬭, *softgoals* ⬭, and *tasks* ⬭. A softgoal differs from a goal in that the former can only be satisfied sufficiently (i. e., actors do not agree on how to measure the satisfaction of a softgoal or a measure does not exist for it at all), while the latter can be satisfied objectively (i. e., there is a standard, agreed-upon way to measure the satisfaction of a goal). Be Efficient is an example of a softgoal, as it is always possible to be more efficient. Be Legally Compliant is an example of a goal, as it is possible to determine whether one is legally compliant or not. Tasks (e. g., Duplicate Detection) often describe operational activities of solutions but may also describe characteristics of solutions such as their speed or resource usage. Intentional elements can have a relative importance to their containing actor, shown as a value between parentheses. Similarly, actors can have a relative importance level as well. A qualitative scale (high, medium, low, none) or a quantitative scale ([0..100]) can be used here. Note that large models can span multiple diagrams.

Intentional elements can be connected to each other using three types of links:

- *Decomposition* links ⟶+ of types AND, OR, or XOR. For example, Duplicate Detection is decomposed into two mutually exclusive alternatives. The satisfaction of the decomposed element is the minimum satisfaction of the decomposing elements for AND, and the maximum for OR and XOR. In XOR's case, only one alternative can be satisfied at a time.
- *Contribution* links ⟶ with a qualitative or quantitative weight (in $[-100..100]$). Fully Automated contributes negatively ($-25$) to Be Effective and positively (25) to Be Efficient. The satisfaction of an element is the weighted sum of products (satisfaction × contribution weight) of the contributors, divided by 100 and truncated to $[0..100]$.
- *Dependency* links ⟶●. For instance, the satisfaction of Be Legally Compliant cannot be higher than that of Support Reviews, upon which it depends.

Unlike most other goal-oriented languages, GRL supports *indicators* ⬭, also called Key Performance Indicators (KPIs), which aim to convert a measured value observed from the external world to a unit-less satisfaction level in the [0..100] scale

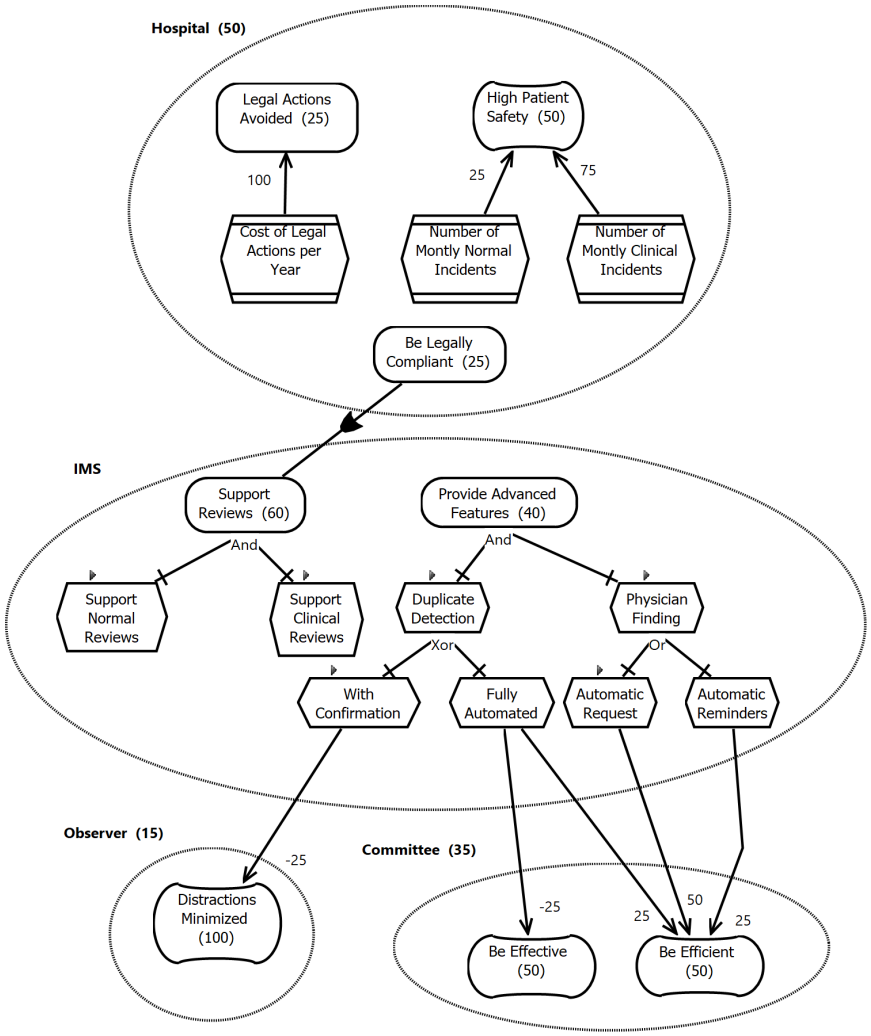



*Figure 3: IMS: Goal model in GRL.*

that can be propagated to the rest of the goal model elements. This conversion uses three indicator parameters, namely the *target* (corresponding to a satisfaction of 100 if the observed value meets it or does better), the *worst* case (satisfaction of 0 if met or worse), and the *threshold* (satisfaction of 50). Linear interpolation is used for values in between. For example, the Cost of Legal Actions per Year indicator in Fig. 3 could have as parameters (in Euros): target = 0; threshold = 100,000; and worst = 1,000,000. In this context, an observed yearly cost of 550,000 Euros would lead to a satisfaction of 25, whereas a yearly cost of 50,000 Euros would result in a satisfaction of 75.

In addition to the quantitative scales discussed earlier for contribution weights and satisfaction levels, the URN standard also supports corresponding *qualitative* scales, shown in Fig. 4. These labels represent combinations of sufficient/insufficient positive/negative qualitative values. Such labels are useful in a context where little information is known about quantities, which is often the case in early requirements engineering activities. Numerical values in quantitative scales are often inferred from historical data or from experts' opinions through consensus-building mechanisms (Akhigbe et al. 2014; Liaskos et al. 2012; Vinay et al. 2014). There are also methods available

to validate GRL models, for example, based on surveys (Hassine and Amyot 2016), and resolve conflicts regarding modelled quantities (Hassine and Amyot 2017). The rest of this paper will use quantitative values in its examples.

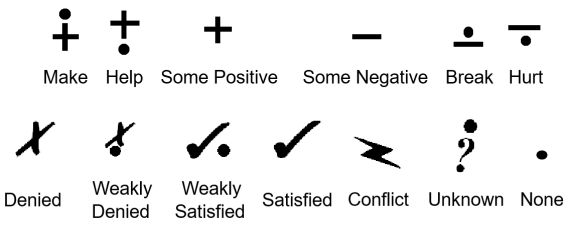



*Figure 4: GRL qualitative labels for contribution weights (top) and satisfaction levels (bottom).*

### 2.2.3 URN Links and Metadata

URN supports user-defined *typed links* that can be used to connect any pair of modelling elements. These are particularly useful to trace elements from the GRL view to those of the UCM view in support of rationale documentation. GRL and UCM elements are annotated with a triangle that indicates the existence of incoming links ◀ or outgoing links ▶. For example, there is a URN link of type *traces* from the GRL task Duplicate Detection in Fig. 3 to the UCM responsibility Check Duplicate in Fig. 1.

URN also supports the tagging of model elements with *metadata*, which are user-defined name-value pairs. The use of metadata and URN links enables profiling URN to specific domains, such as regulations (Ghanavati et al. 2014a). Such concepts can also be exploited during domain-specific analysis.

### 2.2.4 Cognitive Effectiveness

In terms of conceptual coverage, effectiveness, and other such metrics, the UCM and GRL graphical syntaxes fare well compared to similar languages. For example, Abrahão et al. (2019) report empirical evidence that the quality of goal models obtained with GRL is higher than that of *i**, and that GRL is perceived as easier to use and more useful than *i** in several experiments. Mussbacher and Amyot (2008) also assessed the benefits of UCM over BPMN and UML Activity Diagrams in terms of their conceptual coverage of common workflow patterns.

From a cognitive effectiveness however, both GRL and UCM suffer from *symbol deficits* (Genon et al. 2011; Moody et al. 2010). For example, the URN standard does not provide a graphical syntax for i) binding relationships between stubs and plug-in maps, ii) the details of Boolean conditions, iii) scenario definitions, as well as iv) UCM's many performance-related attributes (workloads on start points, probabilities on OR-forks, resource utilisation by responsibilities, etc.) not covered in this paper. See the work of Petriu et al. (2003) for more information on performance aspects in UCM models. On the GRL side, there are also deficits with the absence of concrete syntax for i) strategies, ii) indicator definitions, and iii) advanced concepts such as contribution overrides. URN metadata is not visualized, and typed URN links are not visualized beyond their presence, indicated by triangles (Amyot et al. 2012).

## 2.3 Textual Syntax

A textual syntax for URN models was recently standardized in 2018 (ITU-T 2018), and Kumar and Mussbacher (2018) illustrate its use. This syntax addresses the symbol deficit issues raised in the last section, and enables efficient coding and management of models using standard development environments.

Listing 1 presents an example of the use of the textual syntax with an excerpt of the GRL model from Fig. 3, including the two *softgoals* of the Committee *actor* and two *tasks* of the IMS actor. The textual syntax allows a short name and a full name of an element to be defined (e. g., C and Committee). The short name is used to refer to the element (e. g., in the two *contributions* of the first task of IMS).

Listing 2 presents an example of the use of the textual syntax for the Clinical review *map* from Fig. 2b. At the top, the path is specified, including the *start point* ClinicalReview> and *end point* ReviewDone., *responsibilities* (e. g., "Get Physician"), the *stub* FindPhysician with its *plug-in binding*

specified in parentheses, as well as the *AND-fork* {| with its two branches and the *AND-join* |}.

```
actor C#"Committee" {
   softgoal effv#"Be Effective" {
      importance 50
   }
   softgoal efft#"Be Efficient" {
      importance 50
   }
}

actor IMS#"IMS" {
   task fAut#"Fully Automated" {
      contributesTo C.effv with -25
      contributesTo C.efft with 25
      xor decomposes dDet
   }
   task dDet#"Duplicate Detection" {
      ...
   }
   ...
}
...
```

*Listing 1: IMS: Textual GRL syntax for the IMS goal model (excerpt).*

Connections between path nodes are indicated with ->. At the bottom, the containment of path elements and/or components in a *component* (e. g., Review Committee) is specified.

```
map "Clinical review" {
   ClinicalReview> ->
   "Get Physician" ->
   FindPhysician (
      "Physician finding":
         found=out1,
         PhysicianNeeded=in1) ->
   {|
      "Analyze Report".
      "Analyze Clinical Impact".
   |} ->
   "Categorize Incident" ->
   ReviewDone.

   actor ReviewCommittee:
      ClinicalReview,
      "Get Physician",
      "Analyze Report",
      "Categorize Incident",
      ReviewDone,
      Physician
   actor Physician:
      "Analyze Critical Impact"
   team IMS:
      FindPhysician
}
```

*Listing 2: IMS: Textual UCM syntax for the IMS Clinical review process.*

## 2.4 URN Modelling and Analysis

On one hand, the URN standard itself does not impose a methodology for *creating* models. The first methodology for iteratively creating URN models, including their GRL and UCM views, was proposed by Liu and Yu (2004), with a focus on information systems. One good observation here is that goal elicitation is often easier to do with managers working at a strategic level whereas process elicitation is often more effective with stakeholders closer to the operational level. On the other hand, the URN standard formalizes modelling concepts and guidelines for *analysing* GRL models and UCM models.

The analysis of GRL models is done using *strategies*, which are sets of initial satisfaction values for intentional elements and of observed values for KPIs, and *propagation algorithms* that compute the satisfaction of the other linked elements, their actors, and the entire model. These algorithms can use as input qualitative satisfactions, quantitative satisfactions, or a mix of both (Amyot et al. 2010). The jUCMNav tool, a free Eclipse-based environment for URN modelling and analysis[2] , supports seven such algorithms (Amyot et al. 2012).

For example, Fig. 5 shows the jUCMNav tool with the evaluation result of the GRL model in Fig. 3 for a given strategy, where the With Confirmation and Automatic Request tasks have been selected, and where the three KPIs have been fed observed values. The KPIs first compute their corresponding satisfaction levels as explained earlier, and then the quantitative algorithm used here propagates these initial satisfaction levels to the

[2] https://github.com/JUCMNAV/projetseg-update/wiki

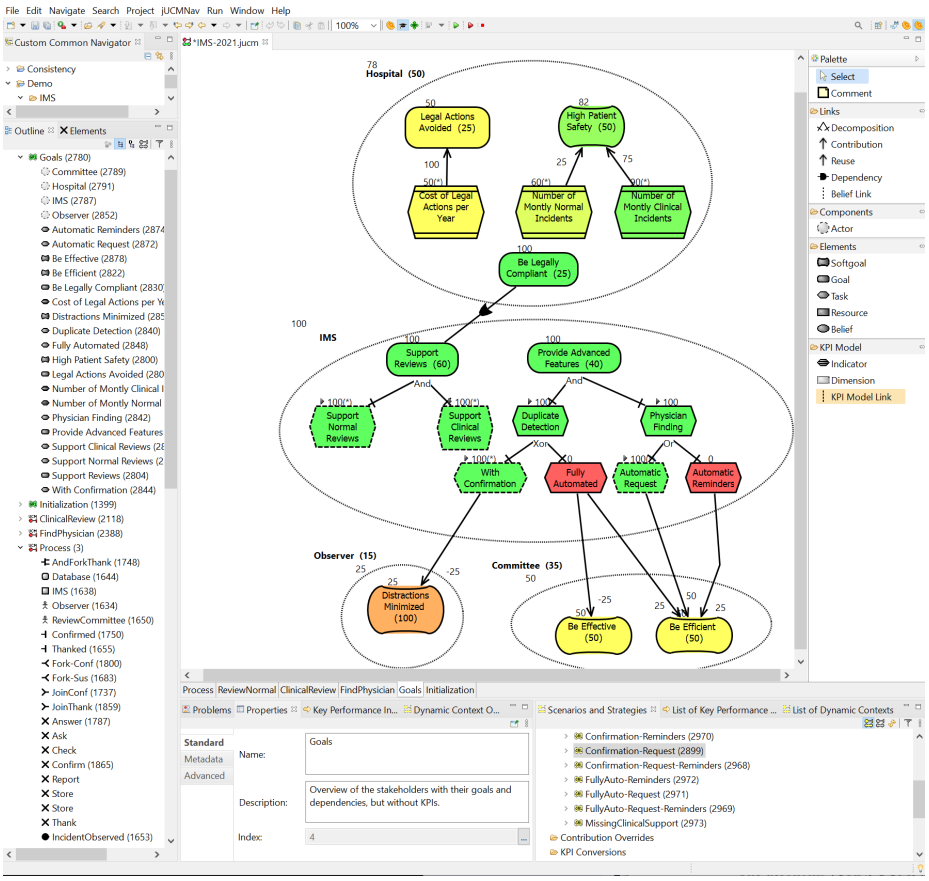



*Figure 5: GRL model, evaluated for one strategy, in the jUCMNav tool.*

other model elements, in a bottom-up way. Colour feedback is also provided: the greener the better, and the redder the worse.

Different strategies can be defined and compared, hence supporting "what-if" analysis as well as trade-off assessments (i. e., which stakeholders will be satisfied or not).

Analysis of goal models can also be done in a top-down way, i. e., to produce a GRL strategy that will optimize some goals given other constraints. For example, Fan et al. (2018) recently provided arithmetic semantics for GRL models and automated the generation of mathematical functions from GRL models, in jUCMNav. These functions can be fed to optimizers such as CPLEX (IBM 2019) in order to find an optimal trade-off, represented in the form of a strategy, given some optimization objectives and constraints (Anda and Amyot 2020). On the UCM side, the URN standard provides the concept of *scenario definition*, which defines the initial context used to traverse a UCM model. A scenario definition specifies which start point(s) are triggered, initial values for the model's variables, expected end points reached at the end, as well as optional pre/post conditions.

A UCM model can include Boolean, Integer, and enumeration variables used to specify guarding conditions (using a syntax based on Java). For example, the IMS model used here (Figs. 1 and 2) contains Boolean variables (DupSusp, DupConf, and Clinical) and Integer variables (Attempt and NumAttempt). UCM responsibilities can also contain code that updates these variables during the traversal. For example, Attempt is set to 0 by responsibility Get Physician and is incremented by 1 by responsibility Ask To Join.

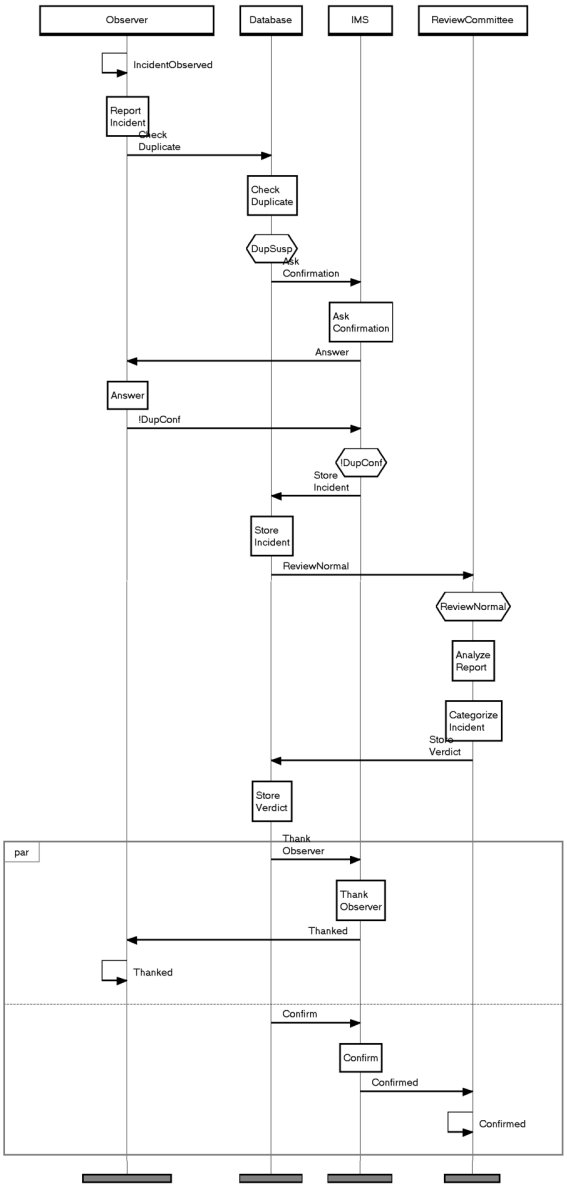



*Figure 6: IMS: Traversal result for a UCM scenario definition, shown as a sequence diagram.*

The path highlighted in red in Figs. 1 and 2 results from the traversal of the model for a scenario definition by jUCMNav. In this definition, the start point is *IncidentObserved* and the initial values specify that there is a duplicate incident suspected (DupSusp=true) but not confirmed (DupConf=false) for a non-clinical incident (Clinical=false). The traversal detects that the normal review sub-process must be used here (i. e., Fig. 2a) and not the clinical review.

The UCM model traversal based on scenario definitions enables testing the process model and avoiding regression during modifications. jUCMNav also enables exporting and visualizing traversal results as sequence diagrams. For example, Fig. 6 corresponds to the flat representation of the path highlighted in red in our example. Such diagrams are helpful to communicate with stakeholders and systems engineers, and even for deriving test cases.

## 3 URN Application Areas

The literature review from (Amyot and Mussbacher 2011) identifies multiple application areas for URN models, from telecommunication services and business models to healthcare systems. Using queries on Google Scholar and Scopus, this section provides an updated perspective with coarse-grained statistics for important clusters of application areas.

In order to get an order of magnitude of the number of URN-related papers published in the past few decades, we ran the following query on Google Scholar (including patents but excluding citations) on March 8, 2021:

"User Requirements Notation" **OR** "Use Case Map" **OR** "Use Case Maps" **OR** "Goal-oriented Requirement Language" **OR** "Rec. Z.151" **OR** "Recommendation Z.151" **OR** jucmnav.

The acronyms (URN, UCM, and GRL) were excluded as they led to too many false positives. However, jUCMNav was included as the main URN modelling environment. This resulted in 3,730 results, including 70 patents. The first papers were from 1995, which is the year the "Use Case Map" expression was coined.

As Google Scholar is known to index many low-quality documents and some grey literature, and as it indexes the full text of the papers (meaning that URN could just be mentioned in passing) a similar query was ran on a multidisciplinary, reputable, and curated search engine. The results presented in the remainder of this paper are based on this

curated search engine and not the Google Scholar search. Elsevier's Scopus[3] was selected here as it offers a flexible query language, it enables searches on various publication metadata (leading to higher precision in the selection of papers that actually focus on URN), and it has good analytics capabilities. The following query was used:

**ALL**( "User Requirements Notation" **OR** "Use Case Map*" **OR** "Goal-oriented Requirement Language" **OR** "Rec. Z.151" **OR** "Recommendation Z.151" **OR** jucmnav )
**AND** ( **EXCLUDE** (DOCTYPE , "cr") )

The last part of this query excludes conference reviews (e. g., prefaces of proceedings). This query on all metadata (but not on full texts) resulted in 1,466 URN-related papers since 1995, from 76 countries. A more restricted version of this query scoped to titles, abstracts, and keywords (i. e., using **TITLE-ABS-KEY** instead of **ALL**) resulted in 305 papers, from 37 countries. These are papers with core contributions to URN or that use URN substantially.

Fig. 7 displays the distribution of these papers, per year. The dotted line (all metadata) uses the vertical axis on the right of the figure whereas all six other lines use the one on the left. The Title/Abstract/Keywords distribution is the one discussed earlier. The five other lines each represent a category of papers where the query based on all metadata was restricted (using **AND**) with additional terms for popular application areas searched in titles, abstracts, and keywords only.[4]

- *Telecom/Networks* (89 papers): Telecommunication features, telephony services, and other networking protocols and applications collectively represent the original area for which URN was developed. URN quickly became popular here, even before its standardization, especially for validation and verification purposes. However, this area slowly faded away as other areas became more popular in the last decade.
- *Software/Enterprise Architecture* (139 papers): One important area concerns the application of URN to software architectures and enterprise architectures, including the reverse-engineering of existing architectures, trade-off analysis, and performance analysis. Interest in this area peaked around 2015 and also seems to slowly fade away.
- *Laws/Regulations* (133 papers): Another popular area, more surprising this time, relates to legal and compliance aspects. Many papers used URN to model laws, regulations, and policies, often using GRL, together with mandatory processes in UCM. Analysis often involved traceability (for change management), compliance analysis, and performance analytics. Interest in using URN for legal aspects has increased substantially since 2009, and has shown the highest level of activity for the past 5 years among the five areas we identified.
- *Business/Process/Value* (110 papers): For the past 15 years, and at a sustained frequency, URN had much success in its application to business process modelling and management (often exploiting the UCM view), but also in business and value modelling (often exploiting the GRL view).
- *Health/Medicine* (71 papers): In the last decade, one increasingly important type of business area targeted by URN publications relates to healthcare and medicine, often from a clinical perspective (e. g., medical procedures), but also from an administrative perspective, for regulatory compliance, process improvement, or change management.

Of course, the above assessment is subject to many limitations as only one search engine was used, and there likely are relevant papers not included. These categories are also not mutually exclusive, i. e., one paper could be part of many categories. Still, the historical trends and proportions are telling and likely representative of reality in the scientific literature.

[3] https://www.scopus.com/

[4] The Scopus queries and raw numbers are available as a companion file in this submission.

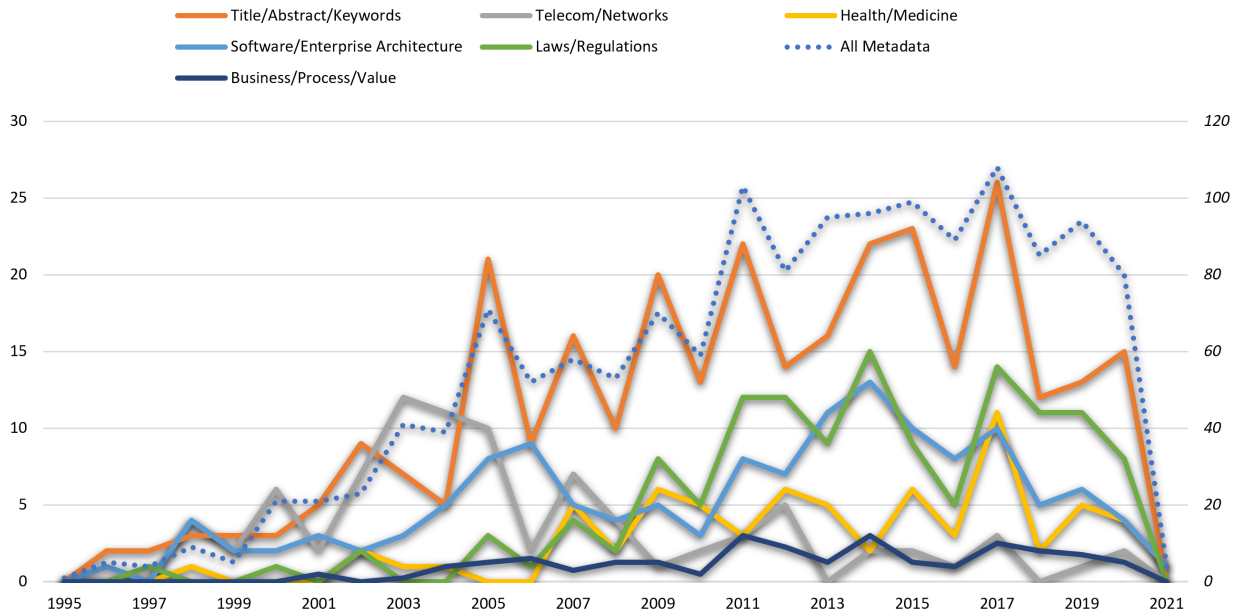



*Figure 7: Number of URN papers per year, for several categories, based on Scopus queries ran on March 8, 2021.*

A closer look at these papers, this time using a Scopus query that looks for titles, abstracts, and keywords that cover *both* UCM and GRL, suggests that only about 10% of the above papers actually address a combination of goal and process modelling. This still concerns a significant number of papers, and many contributions combining goal modelling in GRL with process modelling in UCM will be covered in the next sections.

Interestingly, many *patents* (70 according to Google Scholar) were also influenced by URN. Some simply use UCM models to describe scenarios or processes as part of the patent documentation, e. g., see Liscano et al. (2008) and Mears et al. (2016). Other patents however build their solutions on top of URN, for example by UCM modelling as part of a larger method (Dotan et al. 2013), UCM models as input (Weigert 2017), or GRL models for system configurations (Franchitti 2019). A commercial version of the jUCMNav tool with additional verification and test generation capabilities for UCM models was also produced by Updraft (formerly Uniquesoft, see Weigert et al. (2019)). This shows that URN is used both in academia and in industry.

While this section grouped the papers into application areas such as Telecom/Networks and Health/Medicine, the following six sections offer an orthogonal grouping into key research areas focusing on *combined goal/process modelling* that is based on the literature review as well as the authors' personal experiences.

## 4 Goal/Process Alignment

Organizational alignment is an area that certainly benefits from a combination of process and goal modelling. Whether the existence and definitions of processes meet organization and stakeholder goals, and whether goals are properly covered by such processes are important questions to answer, especially as both goals and processes evolve.

In addition to the work mentioned in the introduction (Cardoso et al. 2011; Koliadis and Ghose 2006), other work in that area includes the approach of Guizzardi and Reis (2015), which provides an informal and manual method for tracing BPMN processes to Tropos goals in order to assess alignment. Sousa and Leite (2014) studied the merging of *i** goals and BPMN processes, and the addition of indicators to trace and measure

alignment. More recently, Insfrán et al. (2017) explored a combination of GRL with BPMN in an approach that promotes traceability and process alignment by construction, with several mapping and prioritization rules. This paper is particularly interesting in its combination of GRL with a process language other than UCM. Behnam and Amyot (2013) also explored the selection and configuration of reusable process patterns in UCM based on GRL goal models evaluated based on a given organizational context.

Most of these approaches attempt to establish traceability between goals and processes at construction time. While there are benefits in ensuring that processes satisfy their objectives when they are built or selected, such approaches are limited in their assessment of alignment when the involved goals and processes evolve over time.

Akhigbe et al. (2016) provided an initial set of consistency and completeness rules that help establish and maintain alignment between GRL goals and UCM processes. Given the importance of a basic infrastructure to assess alignment, this work was extended by i) considering new rules that exploit decomposition structures in these models, ii) automating the verification of these rules in jUCMNav, iii) providing usability features in jUCMNav to simplify the tagging of elements, and iv) providing model transformations for re-establishing alignment in case of rule violations.

Alignment rules are not trivial when balancing correctness with usability, especially for large models. Simple rules requiring that all goals be covered by a process or an activity are simple to implement, but:

1. Not all rules apply the same way to all goal/process models; they must be selectable by modellers.
2. Not all intentional elements in a goal model are operationalized as process model elements (indicators being a good example here). Similarly, not all process elements need a justification captured in a goal model. Such GRL and UCM model elements must not be subject to the alignment rules, to avoid false positives.
3. There are just too many elements to align in real models; some rules must allow taking advantage of model structures (scoped to diagrams, containers, etc.) to imply coverage.
4. Checking the structural presence or absence of certain links is simple, but ensuring that the two aligned model elements are the right one is difficult.

A set of 18 goal-process alignment rules is provided for URN[5] , importable in jUCMNav, to address the first three issues discussed above. The fourth issue remains to be addressed at this time. The rules are formalized as OCL constraints that can be checked selectively against URN models, using the jUCMNav's constraint selection and verification feature (Amyot and Yan 2008).

These rules exploit a specific metadata (*Traces=No*) that indicates, when used to annotate a model element, that this element is outside the scope of the rules. jUCMNav now includes a button to add and remove this annotation in one click. The rules check the presence of a particular type of URN link (see Sect. 2.2.3) of type *traces*. Often, these rules come in pairs, for instance:

- Each GRL actor must have a *Traces* link to a UCM component, unless tagged with *Traces=No*.
- *Traces* links from a GRL actor must only be to a UCM component.

The first one looks for the existence of a UCM component aligned with that GRL actor, whereas the second one restricts the type of links from GRL actors. A similar pair of rules exist from UCM components to GRL actors.

Some rules are also meant to be mutually exclusive, for instance:

- Each GRL intentional element must have a *Traces* link to a UCM map or responsibility, unless tagged with *Traces=No*.

[5] Available at http://bit.ly/URN-alignment-rules

- Each GRL task must have a *Traces* link to a UCM map or responsibility, unless tagged with *Traces=No*.

The first one requires the alignment of all types of GRL intentional elements (goals, softgoals, and tasks) with UCM process elements whereas the second one restricts this alignment to GRL tasks only.

To minimize the number of Traces links that must be manually created by the modeller, some rules infer alignment of finer-grained model elements from the alignment of more coarse-grained elements. In this context, alignment can take advantage of containment relationships (intentional element to actors in GRL, or path elements to component or maps in UCM) and refinement relationships (e. g., decomposition in GRL and stub/plug-in relationships in UCM). Such rules include:

- Each UCM component (or one of its *parents*) must have a *Traces* link from a GRL actor, unless tagged with *Traces=No*.
- Each GRL intentional element (or one of its decomposed *parents*) must have a *Traces* link to a UCM map or responsibility, unless tagged with *Traces=No*.

The above rules are recursive in order to support multiple levels of containment and refinement. Such powerful rules must be used with care as they will cover many sub-elements all at once, including some that might be unjustified.

Fig. 8 shows the result of applying 14 of the 18 alignment rules on the IMS example from Sect. 2. Seven instances of violations of four different rules are reported, together with the rule name and the location in the model. Right-clicking on a warning also enables a contextual menu to offer three typical types of fixes: i) delete the element, ii) create a new element (of the type expected by the rule) with the expected *traces* link, and iii) ignore the element (by adding the *Traces=No* metadata).

In this example, the Hospital GRL actor does not have a corresponding UCM component. This actor should likely be ignored as it is not involved in the process. The GRL task Fully Automated should likely be ignored as well (as it was not selected for this process), whereas the GRL task Automatic Reminder might be kept but with corresponding (and linked) improvements to the UCM process. The UCM component Physician is also not aligned, and one solution would be to augment the goal model with a new corresponding (and linked) actor, with its goals. Such decisions need to be done for the other warnings as well, and alignment reassessed after these modifications. Using a different selection of rules would also lead to a different set of warnings.

There are clear **benefits** here in not only establishing alignment when one view is generated from the other, but in continuously maintaining alignment when any of the goal or process views is modified. Only the rules that make sense need to be checked, and only on the modelling elements that require it. Recursive rules that take advantage of structural aspects such as refinement and decomposition also help minimize manual labour for large models. Tool support where both views are integrated, links can be added at construction time, and rule violations can be fixed in one user action also help with the usability of the alignment maintenance.

## 5 Regulatory Compliance and Intelligence

Regulatory compliance aims to ensure that the business operations of an organization are aligned with relevant laws, regulations, policies, and other such obligations. These business operations can be modelled with business processes, but most regulatory aspects, which are more often based on outcomes than on prescribed behaviour, cannot all be modelled with processes. A recent survey by Hashmi et al. (2018) indicates two important types of regulatory compliance: at design time (which often implies the use of models and traceability) and at run time (which focuses on checking behavioural instances against expected obligations and processes).

*Regulatory Compliance Software Engineering*, which focuses on the development of systematic approaches to build and maintain legally-compliant software systems, is an area of research that benefited from a combination of goal and process modelling. In the last decade, with the introduction of new regulations such as HIPAA (U.S. Government 1996) for US healthcare and GDPR (Intersoft Consulting 2016) for EU privacy, goal modelling languages, and especially GRL, have been extended to help model regulations in the same way as software and business requirements (Ghanavati 2013; Ghanavati et al. 2014a,b) or to help evaluate and establish regulatory compliance.

These approaches build on the idea that modelling regulations and legal requirements in the same notation as other types of requirements will help improve the understanding of regulations, manage compliance with regulations in a systematic way, and identify non-compliant instances in software and processes. Akhigbe et al. (2019) recently reported on a literature review of methods used for legal and regulatory compliance and showed that goal modelling methods, such as those based on GRL, offered more **benefits** for modelling compliance tasks in comparison to non-goal-oriented methods, such as logic-based ones. This is in part because the goal-oriented methods can be used to model both the intent and the structure of laws and regulations, whereas the other methods focus mainly on their intent. Although regulations generally describe only what the intent of compliance is (leaving some freedom for organizations to decide how best to comply in their context), on a few occasions, they also *prescribe* operational aspects, i. e., high-level processes. To capture such operational aspects, researchers also proposed including UCMs in the modelling of regulations and other legal concepts, with traceability links between the GRL and UCM views.

Ghanavati (2013) and Ghanavati et al. (2014a,b) developed a domain-specific version of URN (i. e., a *profile*) within the *Legal-URN* framework, which models legal requirements with Legal-GRL and existing legal processes with Legal-UCM. Fig. 9 shows an overview of Legal-URN. Legal-URN includes three layers where organizational processes/procedures and policies are modelled in UCM and GRL respectively, and where regulations are modelled with Legal-GRL and Legal-UCM. Organizational models (for instance, similar to the IMS example of Sect. 2) and legal models are connected to each other via various types of traceability links. In Legal-URN, the top layer (i. e., procedures & policies and multiple regulations) helps track the models to their source documents. The second layer, with organizational GRL and Legal-GRL

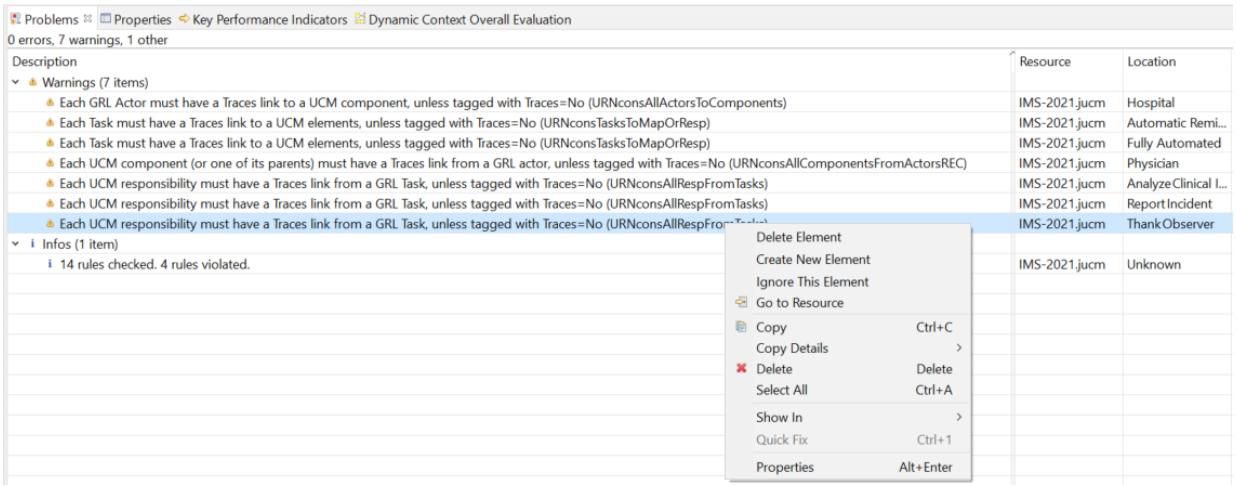



*Figure 8: Warnings reported by jUCMNav for violated alignment rules, with potential fixes.*

models, supports quantitative and qualitative legal compliance analysis. The last layer, composed of organizational UCM and Legal-UCM models, helps ensure business process compliance. Legal-URN enables evaluating legal compliance through GRL *strategies* using tailored qualitative and quantitative propagation algorithms that exploit the new profile information.

Legal-GRL introduces new types of intentional elements for GRL by adding *stereotypes* (described as URN metadata) to softgoals and goals. These stereotypes are based on deontic notions of legal requirements (i. e., permissions and obligations). Softgoals and goals in Legal-GRL are annotated with «Permission» and «Obligation» stereotype metadata. *Obligations* are intentional elements that are mandatory by law ("shall", "must", and "will" are usually used to indicate obligations in legal texts), whereas *permissions* are optional legal requirements (often indicated by the presence of "may" or "might"). Legal-GRL also includes other stereotyped intentional elements to illustrate a «Precondition», an «Exception», or a cross-reference «XRef», together with consequences of non-compliance in legal requirements (specified using contribution links).

The Legal-URN profile is implemented in jUCMNav and includes 18 OCL rules used to verify the well-formedness of Legal-GRL models and 5 OCL rules to identify non-compliant instances between Legal-GRL and GRL. The Legal-URN profile also **benefits** from the relationship between GRL and UCM in that the Legal-GRL intentional elements and their links can be traced to components, stubs, and processes in the Legal-UCM view. To show the traceability between the two types of models, URN links are used. This traceability helps propagate the satisfaction values of intentional elements in Legal-GRL models to components in Legal-UCM models. Traceability of the models to laws/regulations and to organizational policies is handled through third-party requirements management tools, where models can be imported and modifications to linked models and texts tracked (Rahman and Amyot 2014). Legal-URN hence **benefits** process modelling by better ensuring, at design time, that organization processes comply with models of laws (involving both goals and processes) and maintain compliance when changes occur to laws or organization processes and objectives.

The *Variability-Aware Legal-GRL Framework* (Sartoli et al. 2020a,b) is another example of a domain-specific profile of URN that extends GRL and Legal-GRL. This framework helps to analyze compliance of software systems to multiple regulations in the face of run-time variability and environmental changes and provides adaptive privacy. Adaptive privacy is defined by Sartoli et al. (2020a,b) as "capabilities to monitor information transmission to allow response to contextual changes". In this framework, Legal-GRL's intentional elements are combined with *Privacy-Aware Role Based Access Control* (P-RBAC) elements (Ni et al. 2010) through a set of *stereotypes* (described as URN metadata) to ensure capturing new constructs such as purpose, role, and context, and support evaluating a legal model based on different environmental variables. The Variability-Aware Legal-GRL profile includes three profiled types of actors (i. e., ≪Compliance Actor≫, ≪Data Requester≫, and ≪Legal Actor≫). Permission intentional elements from Legal-GRL are connected to ≪context≫ and ≪ purpose≫ GRL tasks through decomposition links. ≪Capability≫ elements are represented as tasks and, finally, ≪data object≫s are represented as resources. This framework **benefits** process modelling by helping processes better comply dynamically with multiple regulations.

Legal-UCM has been also used to help model ambiguities defined by an *Ambiguity Taxonomy* (Massey et al. 2014) in legal statements (Massey et al. 2017). In this work, UCM stubs ◇ are used and stereotyped to cover six categories of ambiguities: lexical, syntactic, semantic, vagueness, incompleteness, and referential. In this work, the legal requirements are categorized as non-ambiguous or ambiguous. If a requirement is non-ambiguous, it is treated similar to system requirements and is modelled as-is. Otherwise, if a requirement is ambiguous, a stub is added in the UCM path

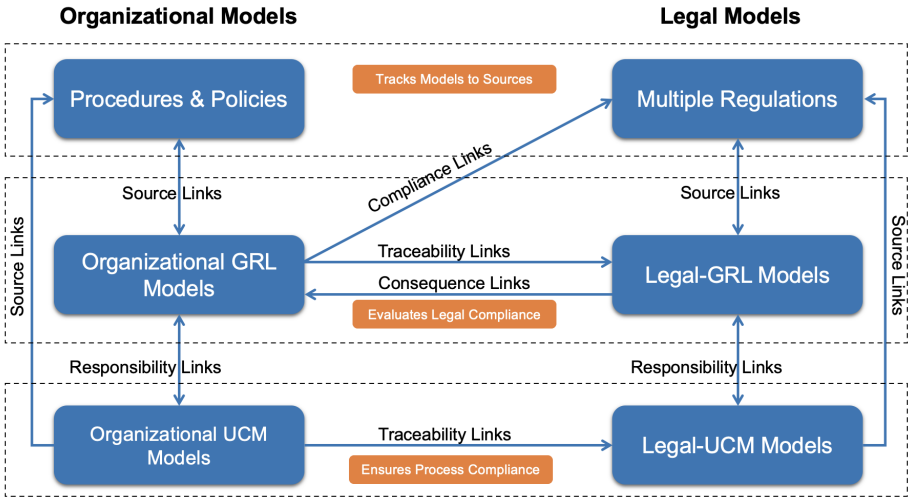



*Figure 9: Overview of the Legal-URN Profile.*

and the type of ambiguity is documented in the stub. Such augmented process models **benefit** legal experts as a means to discuss ambiguities and resolve them.

Whereas the Legal-URN work mainly focuses on design-time compliance, a combination of goal and process modelling was also investigated for a run-time compliance perspective. The particularity here is the focus on GRL not only to specify the regulations and policies with which processes need to comply, but on the use of GRL *indicators* to measure this level of compliance in a non-binary way (Shamsaei et al. 2010). The satisfaction levels of indicators and other goal model elements are also exported to a Business Intelligence tool to support compliance analytics. This led to another compliance profile for URN and an Indicator-based Policy Compliance Framework (Shamsaei 2012) with a core focus on aviation and transportation regulations that **benefits** organizations in detecting hot spots in processes, captured with UCM, regarding run-time regulatory compliance.

Further building on this work, Akhigbe (2018) proposed another URN profile supporting the *Goal-oriented Regulatory Intelligence Method* (GoRIM), this time to help regulators assess whether regulations themselves achieve their desired objectives and accomplish their intended outcomes. GoRIM is a method that facilitates evidence-based decision-making through modelling and data analytics using advanced Business Intelligence tools, a domain referred to as Regulatory Intelligence (Akhigbe et al. 2017). GRL enables modelling three complementary views (regulations, regulatory initiatives used to administer the regulations, and intended social outcomes) with the same language and enables robust analysis within each view and across views. A data analytics tool is then used to explore and analyze data derived from goal models. A combination of goal and process modelling in methods such as GoRIM provides **benefits** to regulators in the form of an intelligent approach to regulatory management, as well as a feedback loop in the use of data for continuous monitoring, assessment, and reporting on efficiency and effectiveness of regulations.

## 6 Process Adaptation and Improvement

The URN standard does not specify how the combined analysis of GRL and UCM views can be done. Early work from Roy et al. (2006), implemented in jUCMNav, enables the automatic creation of UCM Integer variables for each intentional element in the GRL view. Each variable

represents the satisfaction level of its corresponding GRL element. The **benefit** of this approach is that each view can *dynamically* influence the other. Satisfaction levels can be used in UCM conditions (influencing the traversal of the UCM model) and in turn traversed responsibilities can include code that modifies the satisfaction level of goals and other elements. This provides a limited form of *process adaptation* that was exploited for the engineering of Next Generation Networks services (Amyot et al. 2008).

This synergy was further extended with support for indicators and a rule-based reasoning engine (DROOLS) for monitoring context in the work of Vrbaski et al. (2012) on their CARGO approach. The **benefits** here include the modelling of the feedback loops of self-adaptive socio-technical systems, reasoning based on a combination of rules and goal analysis at run-time, and an execution/simulation environment for context-based process adaptation.

The above process adaptation strategies focus mainly on run-time adaptation. However, there is also a need for adapting external processes to the context of a specific organization, especially in a technology adoption scenario. For example, a hospital always strives for improving its services and managing its business and clinical processes for optimal allocation of resources. Technology-based solutions aim to enable improvements and optimization of business processes and delivered care services. However, introducing new technological solutions (for instance, the Incident Management System of Sect. 2) in a large organization such as a hospital, is not a straightforward task as it may cause existing business and clinical processes to be changed. This may in turn disrupt current practices and increase user resistance toward those changes. Hence, the impact of the introduced technology and its dependent processes on stakeholder/organization goals, business objectives, and predefined criteria have to be illustrated and evaluated systematically.

Kuziemsky et al. (2010) addressed this issue by using URN to show the possible impact of healthcare information systems on operational business processes and business goals, following five steps:

- Develop a GRL model and define stakeholders and their dependencies;
- Assign KPIs to each goal in order to measure its satisfaction level;
- Develop a UCM model to specify existing processes, in order to identify the impact point of the new system with regards to tasks, goals, and KPIs;
- Analyze two cases: the current situation with changes introduced by the new system, and the current situation without the proposed system with respect to the tasks and goal;
- Assess the alternatives continuously.

Kuziemsky et al. (2010) applied their framework on a palliative care process, with observed **benefits** regarding the framework's effectiveness in identifying and prioritizing indicators to decide whether to implement or resist the proposed changes.

Along the same line, the *Activity-based Process Integration* (AbPI) framework defines a URN profile for a goal-driven, analysis-enabled process integration framework in healthcare (Baslyman et al. 2017a,b). The aim of this framework is to integrate activities of new processes, such as technology-dependent processes, incrementally into existing business and clinical processes of an organization, to minimize disruption and keep existing goals satisfied or see them improved.

The **benefit** of AbPI's incremental integration is to capture and evaluate the impact of each newly integrated activity on the stakeholder goals so that the best process integration alternative is selected.

The input to the framework (see Fig. 10) is a set of *Process-Goal Models* (PGMs) that each contains a GRL goal model, many UCM process models, and a main concern. PGMs are used to capture current processes and new processes to introduce, with their respective goal models. The output of AbPI is a collection of optimal PGMs (usually with only one element) that satisfy stakeholder goals, performance objectives, and predefined criteria. Interactive integration and

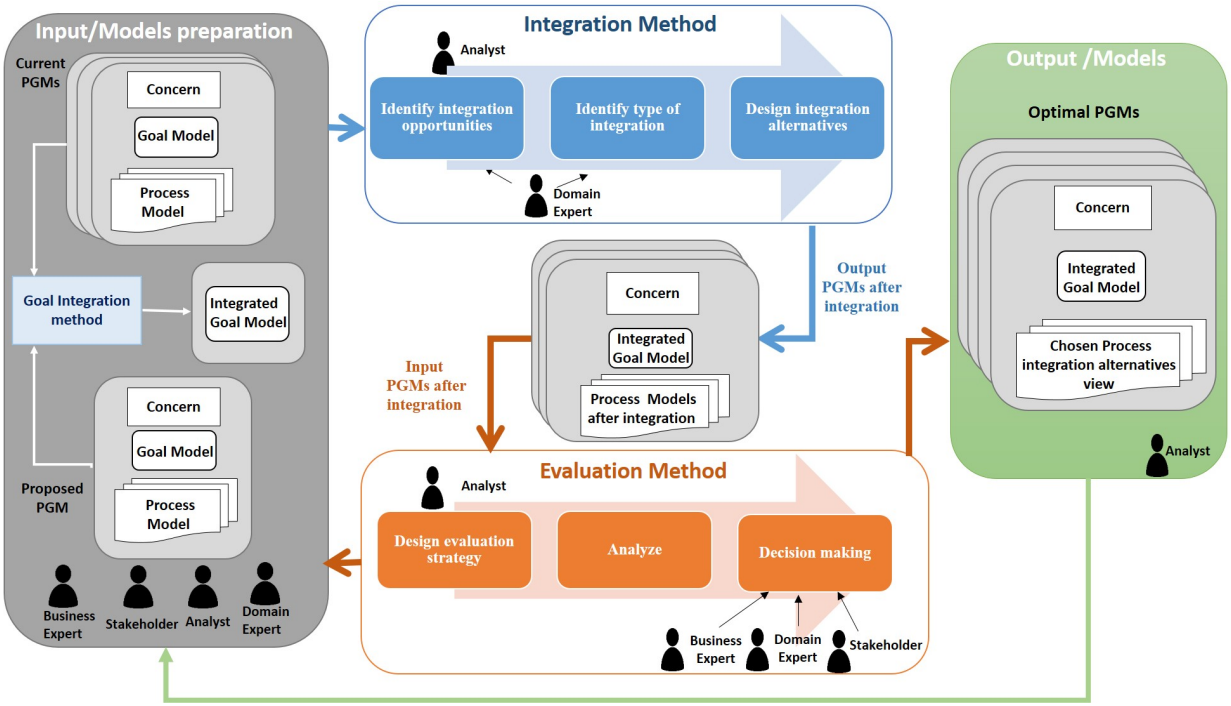



*Figure 10: Activity-based Process Integration framework.*

evaluation methods are used in between to explore and transform the PGMs.

Each activity has a relationship to processes (*current*, *new*, or *integrated*), relationships to other activities of current processes (to *add to*, *combine with*, *replace*, or *eliminate* other activities), and a relationship with the role that performs an activity (*add*, *change*, or *eliminate* the role). In addition, each activity can *contribute* positively or negatively to the satisfaction level of goals or *change* the satisfaction value of goals. These types of relationships were implemented using URN typed links in a URN profile for AbPI. KPIs are used to monitor the performance of process activities of processes and define the impact on the goals. The profile includes OCL constraints to ensure completeness and consistency of the PGM models.

AbPI was also used in combination with the *Lean* change management approach to support effective integration of technology into healthcare practices (Baslyman et al. 2019). The philosophy of *Lean* is to increase productivity while minimizing cost and eliminating waste. *Lean* also uses quantitative measurements to constrain the design of solutions. AbPI can be used in a way that supports the *Lean* approach by providing essential information systematically such as i) the prioritization of goals based on customer values, ii) guidance for the integration of a new process into current processes based on added-value and non-adding-value activities, and iii) quantitative measurements with analysis to reason about integration alternatives and select the best one that meets predefined criteria. An interesting **benefit** here is that goal models help formalize social concerns and measures during the *Lean*-oriented change management of processes.

## 7 Value Co-creation and Service Systems

*Service engineering* has been core to the development of URN (Amyot et al. 2008). It is also a domain in which process modelling and goal modelling have been widely used in the past two decades, but often separately. One of the most well-known, process-based modelling notation for designing and managing services is

*service blueprinting* (Bitner et al. 2008). Service blueprinting illustrates the sequential processes composing a service in terms of physical evidence (e. g., parking), customer action (e. g., give bags), onstage contact activities (e. g., greet), and backstage activities and support processes, including technological systems. Extensions to service blueprinting have added graphical modelling techniques to express actors' goals, among others, in order to better support the analysis of varied service experiences (Patrício et al. 2018). A number of propositions have also used goal modelling to support service analysis and design. For example, *i** has been successfully applied to model recurring business models and patterns in business services (Lo and Yu 2007). It was also adapted to create a modelling technique for expressing the processes by which service actors collaborate to create valued service outcomes, leading to *value co-creation modelling* (Lessard 2015b).

The interest in modelling goals for service design and engineering reflects the importance of analyzing multiple actors' perception of the *value* generated through the service process (Lessard 2015b). Indeed, while a service expressed as a process model would end once the service has been delivered or received, its true endpoint should be the determination of value by its stakeholders once they have integrated it (or not) as a resource within their environment (Lessard et al. 2020). Moreover, goal modelling notations are able to express concepts that are key to the engineering of service systems beyond activities, including agent roles and dependencies in a service network and metrics and indicators of value (Lessard 2015a).

URN enables an integrated analysis of a service system across these dimensions through GRL elements and linked UCM processes. This has been demonstrated through the creation of a lightweight GRL profile and accompanying heuristics for modelling service systems, used to elicit requirements for a telemonitoring service (Lessard et al. 2020). Results showed that requirements identified through the use of the lightweight GRL profile are more complete than comparable requirements proposed in the literature, identifying additional requirements related to the access and use of the telemonitoring solution. Building on the example of the IMS illustrated in Figs. 1 to 3, such an application of URN would enable modelling an incident management *service* in terms of the impact of reviewed incident reports on the hospital, the committee, and other stakeholders' perception of value. This could uncover other important system requirements regarding, for example, information access rights provided to patients about incidents.

In summary, the integration of goal and process modelling has a number of **benefits** for service system engineering:

- The ability to define and measure a service's value from varied stakeholder perspectives, in relation to the processes generating service outputs and outcomes.
- The ability to redefine the endpoint of a service process at the moment when value is determined by stakeholders, in order to capture activities that impact the perceived success of the service.
- The ability to support the identification of additional system requirements that allow organizational actors to integrate a service's outputs and outcomes in a manner that is beneficial to them.

## 8 Goal-oriented Process Mining

Process modelling is often done for future *to-be* processes. However, there is also a need to model *as-is* processes in organizations, ideally based on evidence rather than on biased perceptions of what these processes currently are. *Process mining* is an evidence-based approach to turn event logs into valuable insights about processes (Van Der Aalst 2016). In particular, process mining exploits event logs to discover real processes performed in organizations, enabling modellers to (re)design and improve process models. Process mining activities are categorized into three main types: i) Process Discovery, where a workflow model is inferred from the event logs; ii) Conformance Checking, where the real processes and their instances are compared with the desired or

prescribed models to find and analyze deviations; and iii) Process Enhancement, where the desired data is used to improve or/and extend an existing process model (Van Der Aalst 2016).

The main input of conventional process mining activities is the event log, resulting from the execution of processes. A log minimally includes, for each event, an instance/case identifier, the event name, and timestamps, optionally with other attributes. Process mining techniques have been growing into an activity-focused approach that does not typically consider goals pursued by individual cases and satisfaction levels for different stakeholders' goals (Ghasemi and Amyot 2020). This situation usually threatens the rationality behind the models discovered by process mining methods and often results in unstructured and hard-to-read "spaghetti-like" process models. Although such complex models can reflect reality, they cannot distinguish the traces misaligned with goals from goal-aligned ones (Pesic and Van der Aalst 2006). Such problems, especially in flexible environments that allow many alternatives during process execution, are to be dealt with by process mining practitioners.

A literature review of the intersection of goal modelling and process mining (Ghasemi and Amyot 2020) showed that although both research areas are growing, few studies are conducted at their intersection. Synergetic effects achievable by combining goal modelling and process mining can bring important **benefits** by augmenting the precision, rationality, and interpretability of the discovered process models and eventually improve the satisfaction of process stakeholders.

To address the above research gap, a new approach called *Goal-oriented Processes Mining* (GoPM) was recently proposed by Ghasemi and Amyot (2019b). GoPM, in general, is a process mining approach concerned with both the sequencing of activities and processes' goals and satisfaction indicators. In particular, a *goal-oriented process enhancement and discovery* (GoPED) method has been proposed within the GoPM context. GoPED's log pre-processing step (blue box on the left of Fig. 11) enhances the input event log by extracting traces from sequences of interleaved events and by adding satisfaction levels of corresponding goals and actors to the different traces as additional attributes (Ghasemi and Amyot 2019a). GoPED then exploits these new attributes (red box on the right of Fig. 11) to select the traces that either meet or do not meet certain goal criteria (Ghasemi and Amyot 2019b), before feeding the filtered log to conventional process mining tool and generate a corresponding process model (in BPMN, Petri Nets, UCM, etc.).

For example, for the IMS example of Sect. 2, a trace of activities may reduce legal costs and help satisfy the goal Legal Actions Avoided but may end up with an incorrect assessment of incidents that leads to a higher number of incidents, negatively impacting the goal High Patient Safety. Other traces could lead to opposite impacts. GoPED exploits goal models to manage such conflicting goals and to support trade-off analysis. Such analysis helps find, from existing evidence, the process model promising to satisfy the conflicting goals with a predefined level. In contrast to conventional process mining that uses the whole event logs to discover a model, GoPM aims to use only a subset of the event log that has satisfied predefined goal-related criteria. With GoPED, good past experiences will be found within the log to be used as a basis for discovering good models for the future and bad experiences will be found to be avoided.

The goodness of traces (evidence) and models is defined using three categories of goal-related criteria: i) satisfaction of individual cases in terms of some goals (case perspective), ii) overall satisfaction of some goals rather than every single case (goal perspective), and iii) a comprehensive satisfaction level for all goals over all cases (organization perspective). Finding the subset of cases that satisfy a minimum satisfaction level from each of the above perspectives is not trivial. This is because adding a trace to the subset might help satisfy the criterion related to one goal and, at the same time, might harm satisfying the criterion related to other goals. In GoPED, Ghasemi and Amyot (2019b) proposed three algorithms

(two of which require constraint-based optimization) for the selection of cases out of enhanced logs from the three above viewpoints. Recent experiments demonstrated their feasibility and scalability. GoPM is expected to guide process mining approaches towards specific goal-related properties of interest.

The **benefits** of goal-oriented process mining hence extend beyond discovering process models from evidence. Adding goal models in a process mining context enables GoPM to discover simpler and understandable process models that meet measurable goal-oriented criteria. In turn, these models can help organizations understand and promote good practices, and help people avoid process options that lead to underperforming trade-offs.

## 9 Other Advanced Modelling Techniques

This section highlights other advanced modelling and analysis techniques with features, aspects, slicing, and concepts from social sciences. They all combine GRL with UCM in a way that complements the previous approaches, and they also benefit from tool support in jUCMNav.

### 9.1 URN and Feature Models

In jUCMNav, URN modelling was augmented with *feature modelling* to benefit from a richer description of relationships among features (with a correspondence to tasks in GRL) and a combined evaluation of feature and goal models with a single evaluation mechanism (Liu et al. 2014). For example, the goal model for the IMS actor in Fig. 3 shows the relationships among tasks with decompositions in addition to the impacts of those tasks on high-level goals. However, the modelling of relationships among tasks is limited in goal models and distracts from the core purpose of goal models, i. e., *understanding* the impact of tasks. A feature model offers optional, mandatory, requires, and conflicts relationships in addition to decomposition with OR and XOR groups. To **benefit** from improved *separation of concerns*, relationships among features (i. e., tasks) are modelled more expressively with feature models, the goal model then can focus on the impacts of tasks, and process (workflow) models describe tasks in more detail. To unify the evaluation, a feature model is considered a *special case* of a goal model. Optional and mandatory relationships are interpreted as contributions with specific contribution values that are aligned with the semantics of feature models. Requires and conflicts relationships are integrated into URN with OCL constraints.

Building on the combined modelling of features and goals and a survey about reusability in goal modelling demonstrating gaps in the support of reuse in goal models (Duran and Mussbacher 2019), goals models with relative contribution values were then applied to decide which features of a reusable artifact are to be selected to satisfy high-level goals (Duran 2018). While many kinds of reusable artifacts exist with clearly defined interfaces, these interfaces focus on structural and behavioural aspects and not the reasons *why* an artifact should be reused. These reasons are captured with goal models in a new kind of interface for reusable artifacts (Duran 2018). However, constraints are imposed on tasks in (GRL) goal

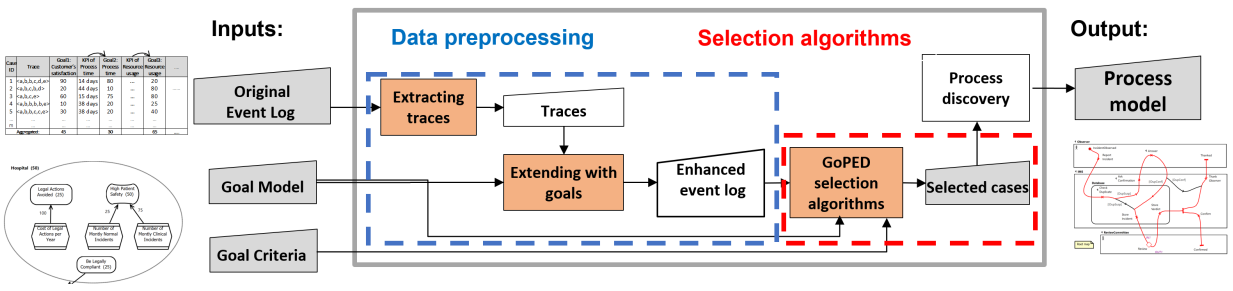



*Figure 11: Overview of goal-oriented process mining, including the GoPED method.*

models not only by feature models but also by (UCM) process models. While a feature model describes which tasks in a goal model may appear together in a system, the process model describes additional sequencing constraints on the tasks in the goal model that have to be taken into account during goal model evaluation (Duran and Mussbacher 2016). For example, tasks that always appear on alternative branches in the process model cannot impact high-level goals at the same time even though both tasks are selected in the feature model for the system. In this case, the *accuracy* of goal model analysis **benefits** from the integration of goal models with feature and process models. These analysis capabilities were then further extended with support for top-down evaluation of goal models (Duran and Mussbacher 2018) that determines the best reusable options (features/business processes) given desired satisfactions of high-level goals. Issues of computational complexity common in top-down evaluation is tackled by exploiting the modularization provided by reuse hierarchies.

### 9.2 Aspect-oriented URN

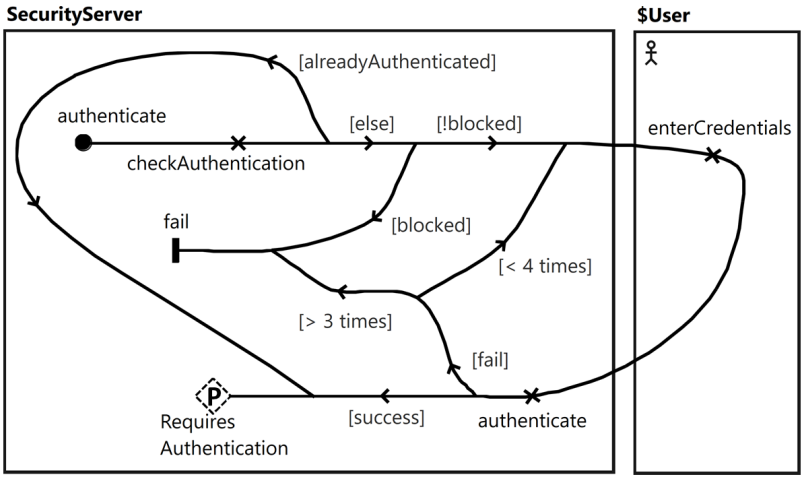
Aspect-oriented modelling allows crosscutting concerns to be encapsulated in aspects, which are then composed to produce an overall model. A crosscutting concern affects many different elements in a model (e. g., an authentication concern needs to be applied to each interaction of a user with the system to ensure security). Aspects generally improve separation of concerns and reusability of modelling artefacts (Pinciroli et al. 2020).

AoURN (Mussbacher 2010), an *aspect-oriented* extension of URN, further enhanced separation of concerns for stakeholders, use cases/processes, and non-functional requirements/system qualities. Integrated goal and process models **benefit** from AoURN as aspect-oriented techniques are applied to both models (Mussbacher et al. 2010a) and *semantic equivalences* are exploited in goal and process models when composing aspectual URN models (Mussbacher et al. 2013). Aspect-oriented techniques require a composition specification that describes where an aspectual model is to be applied. When a model is changed, the composition specifications often have to be changed, too, so that the aspects are still applied correctly. The semantic-based approach of AoURN allows refactoring operations to be performed on an AoURN model without the need to update composition specifications, i. e., AoURN is *refactoring-safe* (Mussbacher et al. 2009). Furthermore, semantic-based interactions among aspects can be detected (Mussbacher et al. 2010b). AoURN is also the basis of an approach based on Software Product Lines that captures features, goals, and processes in a unified framework to reason about stakeholders' needs and perform trade-off analyses while considering undesirable interactions that are not obvious from the feature model (Mussbacher et al. 2012).

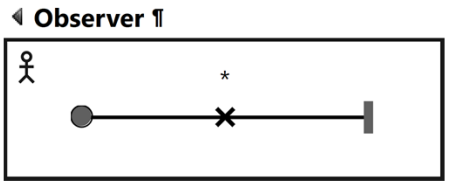
Fig. 12 illustrates the application of an Authentication aspect to the IMS example. The Security Server checks whether a User is already authenticated or blocked, and if not asks the User to enter credentials. If the credentials are correctly entered within three attempts, authentication is successful. The composition specification identifies the responsibilities of the Observer as the target of the Authentication aspect. Although not shown in the figure, the complete composition specification applies the aspect also to the Review Committee and the Physician. When the aspect is applied to the IMS, the Security softgoal with an impact from the Authentication task is added to the goal model of the IMS and an *aspect stub* ◆ is added before each responsibility of the Observer in the IMS root map, clearly demonstrating the crosscutting nature of the aspect.

### 9.3 URN and Slicing

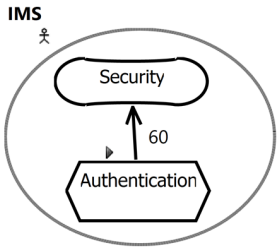
Similar to aspect-oriented techniques that seek to handle complexity through separation of concerns, *slicing* techniques help analysts understand complex models by identifying relevant model elements related to a given element of interest (Androutsopoulos et al. 2013). As UCM process models evolve and grow over time, they rapidly



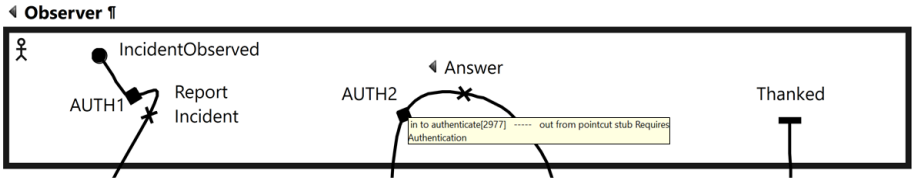

*(a) Authentication Aspect (UCM).*




*(b) Composition specification for Authentication Aspect (UCM).*




*(c) Authentication Aspect (Goals).*




*(d) Authentication Aspect applied to IMS root map.*

*Figure 12: Example of Aspect-oriented User Requirements Notation (AoURN) applied to IMS.*

become hard to understand and to maintain. Indeed, a UCM model may integrate dozens of maps that are themselves composed of hundreds of constructs, which make such amount of information humanly unmanageable even by URN experts. To help analysts understand large and complex UCM specifications prior to performing a maintenance task, Binalialhag et al. (2018) proposed a *static slicing* approach for the UCM notation, implemented in the jUCMNav tool. The technique aims to help requirements engineers reduce UCM specifications according to a *slicing criterion* (composed of a UCM construct and a variable) of interest. The slicing process consists of a backward traversal of UCM path constructs and tracking dependencies to identify path elements within a UCM model that may impact the slicing criterion. The proposed approach produces both closure and reduced slices. A closure slice is displayed by marking the relevant constructs and paths within the original UCM, while a reduced slice is a new executable UCM model obtained after the removal of irrelevant constructs, paths, components, and scenarios.

Fig. 13 illustrates the application of static slicing to the root map of the IMS example, with

responsibility Store Verdict chosen as slicing criterion. The closure slice (colored in green) identifies all elements that may impact the execution of the slicing criterion. In addition, all elements of the other three sub-maps of Fig. 2 are also part of the closure slice.

Later, Alkaf et al. (2019) proposed a forward slicing technique to capture and grasp how changes propagate within UCM models, and between GRL and UCM models through URN traceability links. The developed technique **benefits** maintainers when locating the impact of potential modifications on both GRL and UCM model elements in anticipation of carrying out a maintenance task by increasing their productivity and cutting down the cost of maintenance. Fig. 14 illustrates the application of *change impact analysis* (CIA) to the IMS example, with responsibility Store Verdict chosen as slicing criterion. Potentially impacted model elements are marked in green.

In addition, Fig. 15 illustrates the marked GRL constructs (in purple color, as per the jUCMNav CIA feature implementation) when we apply CIA with task Fully Automated as slicing criterion. The four impacted GRL strategies are also identified (shown as URN comment). The Hospital actor is not shown here as none of its elements are impacted. This support of GRL and UCM elements as slicing criteria, with tool support for forward and backward slicing, bring **benefits** to modellers who need to understand complex process models linked to goal models, and enables them to assess the impact of model modifications before they are implemented.

### 9.4 URN and Social Sciences Techniques

URN has also been integrated with techniques and concepts from social sciences such as psychological paradigms and human values. Information collected based on two psychological paradigms and elicitation techniques, *Activity Theory* (Georg et al. 2015) and *Distributed Cognition* (Hundal and Mussbacher 2018), is systematically transformed into URN models for further analysis. Both **benefit** from the combined modelling of goals and processes in URN to fully describe and consequently analyze the elicited findings, and the explicit consideration of social science paradigms helps discover requirements and process elements that would be difficult to infer otherwise. The Continual Value(s) Assessment (CVA) framework (Perera et al. 2020) again uses goal and process models but also feature models to systematically integrate impact analysis on human values throughout software development.

## 10 Research Challenges and Opportunities

This section discusses recent progress related to generic URN-related challenges identified in the literature a decade ago, and then discusses current research challenges and opportunities related to goal/process integration with URN.

### 10.1 A Decade of Improvements

In their literature review, Amyot and Mussbacher (2011) highlighted eight research challenges for the User Requirements Notation:

1. *Domain-Specific Profiles*: the ability to tailor URN to specific domains was explored substantially in the past decade with different profiles in the legal domain, as discussed in Sect. 5 with the Legal-GRL profile (and its Variability-Aware variant), the Indicator-based Policy Compliance Framework, and the Goal-oriented Regulatory Intelligence Method. Other domains where new profiles were developed include healthcare with the Activity-based Process Integration framework (Sect. 6) and value co-creation in service systems (Sect. 7). Although tool support in jUCMNav enables stereotypes for concepts and relationships, well-formedness constraints, and analysis algorithms to be tailored in profiles, the symbols and terminology used are still based on the URN standard, so part of the initial challenge remains.
2. *Enhanced Workflow Executions*: The 2012 version of the URN standard included many enhancements to the requirements for workflow execution (called UCM *path traversal* mechanism), including new types of dynamic stubs

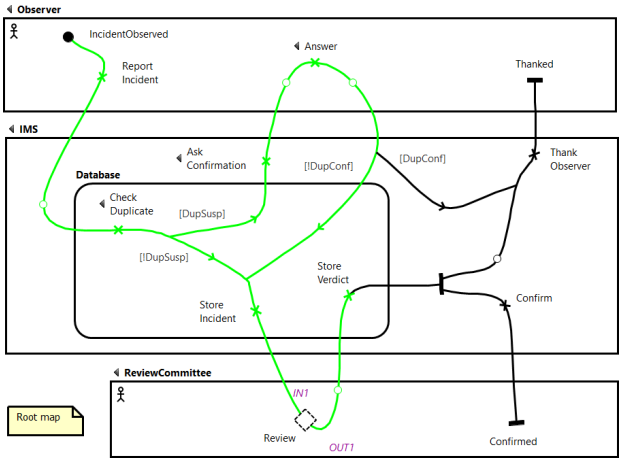



*Figure 13: Closure slice of the IMS root map with respect to the slicing criterion SC = (Store Verdict, -).*

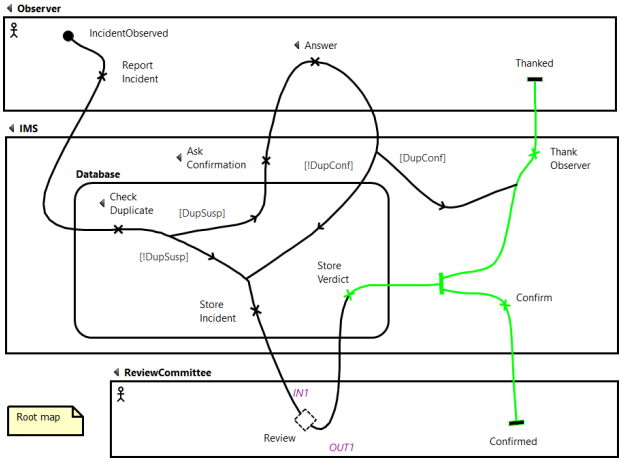



*Figure 14: Marked IMS root map with respect to the slicing criterion SC = (Store Verdict, -).*

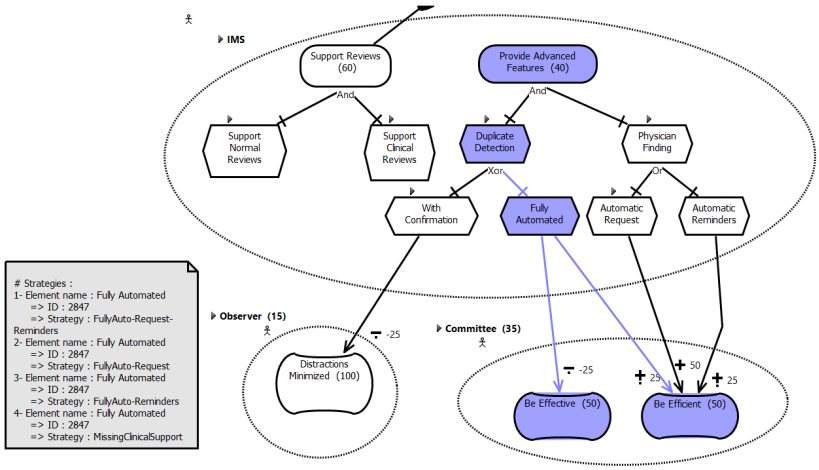



*Figure 15: IMS: Marked GRL elements (extract).*

(with synchronization and blocking behaviour), precise handling of map instances, new types of timers (transient/persistent), and support for exceptions. Many of these improvements made their way to jUCMNav, but a proper debugging environment is still missing. Non-standard execution mechanisms involving aspect-orientation (Sect. 9.2) and slicing (Sect. 9.3) were also explored, with tool support.

3. *Performance Management*: On the GRL side, indicators were added to the standard in 2012 to support performance management in the goal view, with support in jUCMNav. Other non-standard tool features include support for qualitative KPIs, trend computation, and various export and reporting facilities (Amyot et al. 2012). This represented a major advancement to goal-oriented modelling languages.
4. *Compliance management*: There are major results in this area, as highlighted by the publication trends observed in Fig.7. Many such results are discussed in Sect. 5 for legal and regulatory compliance of business processes and of organizations, but also in Sect. 4 in terms of alignment between processes and organizational objectives, as well as in Sect. 6 for business and clinical process adaptation contexts.
5. *Formal Semantics*: Although the URN standard does not include formal semantics, one based on arithmetic functions was introduced for GRL (Fan et al. 2018). Some new formal semantics were also introduced for UCM beyond those that previously existed (covered in Sect. 2.1), e. g., based on Object-Z (Dongmo and Van der Poll 2014) or Maude (Mokhati and Menassel 2013). On the UCM side, these formal semantics still remain partial however as they do not capture all the concepts and informal workflow semantics of the standard.
6. *Improved Analysis Techniques*: There were major contributions in this area, especially in terms of goal model analysis with support for indicators and constraint-based optimizations, support for enhanced workflow executions (point 2 above), analysis techniques exploiting domain-specific profiles (point 1 above), and other combined UCM/GRL analysis techniques highlighted in Sections 4 to 9.
7. *Improved Model Representations*: Although the graphical syntax of URN has not really evolved in the past decade, one major addition in the 2018 edition of the URN standard is a textual syntax (see Sect. 2.3) enabling the coding of models using conventional text-based environments.
8. *Guidelines and Methodologies*: The URN standard focuses on the description of the language rather than on methodological aspects. This general lack of guidelines was addressed by most of the frameworks discussed in Sections 5 to 8, which describe how to use URN or one of its domain-specific profiles to support conceptual modelling and analysis activities.

As shown above, many generic URN-related challenges were partially addressed by the community in the last decade. However, there are new ones that have appeared since, especially around business process/goal integration.

## 10.2 Remaining Challenges and Opportunities

Although much progress has been made in enabling and supporting the combined used of process modelling and goal modelling with the User Requirements Notation, many research challenges and opportunities remain. Each of the areas covered in the preceding sections are ongoing, and research challenges are detailed in their respective referenced sources.

In general, while URN has seen many applications in academia and in industry, further work and education is required to gain wider mainstream adoption by practitioners. One approach would be to further customize UCM and GRL to the needs of specific domains (e. g., financial industry) so that domain-specific terminology and syntax can be used, and reusable assets can be developed. This would go beyond the labeling-based profiles

developed so far, and require better tool support than what is currently offered in jUCMNav.

Some challenges are shared with other goal modelling languages. Scalability, i. e., the ability to handle large models, is a common issue in goal modelling, and can potentially be addressed by language design (e. g., use of aspect-oriented constructs), in the usage methods, as well as by innovative tool support. Although jUCMNav allows GRL models to include many views as separate diagrams and to generate new diagrams from model elements by expanding neighbouring elements, modularity of goal models is still an issue. For URN, the loose coupling of two modelling paradigms also presents some additional research challenges and opportunities.

The primarily manual task of coordinating and maintaining traceability between goals and processes, which is at the core of many of the above techniques for alignment, compliance, adaptation, and value co-creation, can be time-consuming and error-prone. Automating some aspects of these tasks (e. g., identifying goal/process relations through natural language processing (Aung et al. 2020), and checking and maintaining semantic correctness and consistency) would boost productivity and facilitate practical adoption.

Automation of many manual activities beyond traceability is also an important research direction, especially when usability and integration into existing methods used by practitioners must be taken into consideration.

On the methods side, there is potential synergy between URN modelling and performance management methods, as the former can tap into existing KPI definition and data collection processes, whereas the latter can benefit from the analytical and reasoning power of URN to better enable decision making and consensus building.

One significantly new research direction for URN is to provide support for processes that engage with fluid social networks and evolving ecosystems, as well as with socio-cyber-physical systems that include multi-channel heterogeneous content from communities with shifting identities and disparate value systems, as human and automated systems react to each other within ever shortening time frames, leaving little time for rational decision making.

Another significant research direction will be to consider how to integrate the use of advanced machine learning algorithms (including deep learning) that draw on massive data sets and diverse data streams (from the cloud, sensors, and social and mobile networks) and could enhance the current business intelligence research in the areas of regulatory compliance and business process adaptation/improvement.

On a broader scope, as digitalization is rapidly transforming every business sector and almost all aspects of life, we expect that the past two decades of experiences with URN will provide the foundations for addressing many of the challenges that we face today. Processes will need to be more dynamic and agile to adapt to rapid and continual change. Associating and aligning process design with goal reasoning, as supported by URN, provides a foundational building block for model-based transformation.

## 11 Conclusion

One century after the seminal work of Gilbreth and Gilbreth (1921), process modelling is more popular and diverse than ever. Goal modelling, although much younger, has shown interesting potential in complementing process modelling for many important types of activities.

This paper gives a contemporary overview of the User Requirements Notation standard, of its general usage, and of the benefits brought by its combination of GRL and UCM modelling. Although many combinations of existing goal and process modelling languages have been the topic of research activities and practical applications, few combinations have been as well studied in diverse contexts as GRL combined with UCM. Sect. 3 mapped the academic interest in URN modelling and the many application areas where URN was used over time.

The paper answers the posed research question "*What are the benefits of combining goal modelling with process modelling?*" by examining over two decades of contributions of URN-based techniques to the domains of goal/process alignment, regulatory compliance and intelligence, process adaptation and improvement, value co-creation and service systems, and goal-oriented process mining. Advanced URN-based modelling and analysis techniques for features, aspects, slicing, and concepts from social sciences are also discussed.

As a summary of observed benefits, combined goal and process modelling in URN helps in the following ways:

- For the *alignment of goals and processes* (Sect. 4), this combination helps determine i) whether existing processes as they are defined meet organization and stakeholder goals and ii) whether goals are properly covered by such processes even when both goals and processes evolve.
- Regarding *regulatory compliance and intelligence* (Sect. 5), a goal/process combination helps model both the intent and the structure of laws and regulations as well as operational aspects (high-level processes) i) to better ensure that organization processes comply with laws, ii) to maintain compliance when changes occur to laws or organization processes and objectives, iii) to allow experts to discuss ambiguities and resolve them, iv) to detect hotspots in processes regarding run-time regulatory compliance, and v) to manage regulations in terms of continuous monitoring, assessment, and reporting on their efficiency and effectiveness.
- For *process adaptation and improvement* (Sect. 6), we observe that goal/process modelling helps i) capture the dynamic influence of one view on the other, ii) model feedback loops of self-adaptive socio-technical systems based on rules, run-time goal analysis, and a process execution/simulation environment, iii) capture and incrementally evaluate the impact of each newly integrated activity on stakeholder goals to select the best process integration alternative, and iv) formalize social concerns and measures during change management of processes.
- For *value co-creation and service systems* (Sect. 7), this combination helps i) define and measure a service's value from varied stakeholder perspectives in relation to processes, ii) redefine the endpoint of a service process as the moment when it creates value for stakeholders, and iii) support the identification of additional system requirements that allow organizational actors to successfully integrate a service.
- For *goal-oriented process mining* (Sect. 8), combined goal and process modelling helps improve the precision, rationality, and interpretability of the process models discovered through process mining by extracting simpler and understandable process models that meet measurable goal-oriented criteria and eventually improve the satisfaction of process stakeholders.
- Regarding more *advanced URN-based modelling techniques* (Sect. 9):
  - Support for *features and aspects* in combined goal and process models improves i) the accuracy of goal model analysis on which the selection of (potentially reusable) features and processes is based and ii) the coordinated, semantics-based reuse across different types of models.
  - Support for *slicing* in combined goal/process modelling helps modellers who need to understand complex process models linked to goal models, and enables them to assess the impact of model modifications before implementation.
  - This combination also supports the more complete description and subsequent analysis of concepts from *social sciences* (e. g., information resulting from Activity Theory and Distributed Cognition, or based on human values), which in turn promote the discovery of requirements and process elements that would be difficult to infer otherwise.

Although some of the above benefits might be unique to URN, we suspect that many benefits can be generalized to other combinations of goal and process modelling languages. Modelling with two complementary notations does not only bring benefits. In addition to the extra effort required to combine two such partial but complementary views, important challenges and exciting opportunities for future research are discussed in Sect. 10, focusing on the areas of adoption, usable automation, data-driven applications (e.g, AI/machine learning), and socio-cyber-physical systems.

## References

Abrahão S., Insfrán E., González-Ladrón-de-Guevara F., Fernández-Diego M., Cano-Genoves C., de Oliveira R. P. (2019) Assessing the effectiveness of goal-oriented modeling languages: A family of experiments. In: Information and Software Technology 116, p. 106171

Akhigbe O., Alhaj M., Amyot D., Badreddin O., Braun E., Cartwright N., Richards G., Mussbacher G. (2014) Creating Quantitative Goal Models: Governmental Experience. In: Conceptual Modeling: 33rd International Conference, ER 2014. Springer, pp. 466–473

Akhigbe O., Amyot D., Anda A. A., Lessard L., Xiao D. (2016) Consistency Analysis for User Requirements Notation Models. In: Proceedings of the Ninth International i* Workshop (iStar 2016). CEUR-WS Vol. 1674, pp. 43–48

Akhigbe O., Amyot D., Richards G. (2019) A systematic literature mapping of goal and non-goal modelling methods for legal and regulatory compliance. In: Requirements Engineering 24(4), pp. 459–481

Akhigbe O., Heap S., Islam S., Amyot D., Mylopoulos J. (2017) Goal-Oriented Regulatory Intelligence: How Can Watson Analytics Help? In: Conceptual Modeling (ER 2017). Lecture Notes in Computer Science Vol. 10650. Springer, pp. 77–91

Akhigbe O. S. (2018) A Goal-Oriented Method for Regulatory Intelligence. PhD thesis, University of Ottawa, Canada

Alkaf H. S., Hassine J., Binalialhag T., Amyot D. (2019) An automated change impact analysis approach for User Requirements Notation models. In: Journal of Systems and Software 157

Amyot D., Becha H., Bræk R., Rossebø J. E. Y. (2008) Next generation service engineering. In: 2008 First ITU-T Kaleidoscope Academic Conference. IEEE, pp. 195–202

Amyot D., Ghanavati S., Horkoff J., Mussbacher G., Peyton L., Yu E. (2010) Evaluating goal models within the goal-oriented requirement language. In: International Journal of Intelligent Systems 25 (8), pp. 841–877

Amyot D., Logrippo L. (2000) Use Case Maps and LOTOS for the prototyping and validation of a mobile group call system. In: Computer Communications 23(12), pp. 1135–1157

Amyot D., Mussbacher G. (2011) User Requirements Notation: the first ten years, the next ten years. In: Journal of Software (JSW) 6(5), pp. 747–768

Amyot D., Shamsaei A., Kealey J., Tremblay E., Miga A., Mussbacher G., Alhaj M., Tawhid R., Braun E., Cartwright N. (2012) Towards Advanced Goal Model Analysis with jUCMNav. In: Advances in Conceptual Modeling. Springer, pp. 201–210

Amyot D., Yan J. B. (2008) Flexible Verification of User-Defined Semantic Constraints in Modelling Tools. In: Proceedings of CASCON 2008. ACM, pp. 81–95

Anda A., Amyot D. (2020) An Optimization Modeling Method for Adaptive Systems Based on Goal and Feature Models. In: Proceedings of the Tenth International Model-Driven Requirements Engineering Workshop (MoDRE). IEEE, pp. 10–20

Androutsopoulos K., Clark D., Harman M., Krinke J., Tratt L. (2013) State-Based Model Slicing: A Survey. In: ACM Computing Surveys 45(4), pp. 1–36

Antón A. I. (1996) Goal-based requirements analysis. In: Proceedings of the Second International Conference on Requirements Engineering. IEEE, pp. 136–144

Aung T. W. W., Huo H., Sui Y. (2020) A Literature Review of Automatic Traceability Links Recovery for Software Change Impact Analysis. In: 28th International Conference on Program Comprehension (ICPC 2020). ACM, pp. 14–24

Baslyman M., Almoaber B., Amyot D., Bouattane E. M. (2017a) Activity-based Process Integration in Healthcare with the User Requirements Notation. In: International Conference on E-Technologies. Springer, pp. 151–169

Baslyman M., Almoaber B., Amyot D., Bouattane E. M. (2017b) Using Goals and Indicators for Activity-based Process Integration in Healthcare. In: Procedia Computer Science 113, pp. 318–325

Baslyman M., Amyot D., Alshalahi Y. (2019) Lean healthcare processes: effective technology integration and comprehensive decision support using requirements engineering methods. In: 2019 IEEE/ACM 1st International Workshop on Software Engineering for Healthcare (SEH). IEEE, pp. 37–44

Behnam S. A., Amyot D. (2013) Evolution mechanisms for goal-driven pattern families used in business process modelling. In: International Journal of Electronic Business 10(3), pp. 254–291

Binalialhag T., Hassine J., Amyot D. (2018) Static slicing of Use Case Maps requirements models. In: Software & Systems Modeling 18(4), pp. 2465–2505

Bitner M. J., Ostrom A. L., Morgan F. N. (2008) Service Blueprinting: A Practical Technique for Service Innovation. In: California Management Review 50(3), pp. 66–94

Bresciani P., Perini A., Giorgini P., Giunchiglia F., Mylopoulos J. (2004) Tropos: An agent-oriented software development methodology. In: Autonomous Agents and Multi-Agent Systems 8(3), pp. 203–236

Buhr R. J. A., Casselman R. S. (1995) Use Case Maps for Object-Oriented Systems. Prentice-Hall

Buhr R. J. A. (1998) Use case maps as architectural entities for complex systems. In: IEEE Transactions on Software Engineering 24(12), pp. 1131–1155

Cardoso E. C. S., Guizzardi R. S. S., Almeida J. P. A. (2011) Aligning goal analysis and business process modelling: a case study in healthcare. In: International Journal of Business Process Integration and Management 5(2), pp. 144–158

Chung L., Nixon B. A., Yu E., Mylopoulos J. (2000) Non-functional requirements in software engineering Vol. 5. Springer

Clausing D., Hauser J. R. (1988) The house of quality. In: Harvard Business Review 66(3), pp. 63–73

Dalpiaz F., Franch X., Horkoff J. (2016) iStar 2.0 Language Guide. In: CoRR

Decreus K., Snoeck M., Poels G. (2009) Practical Challenges for Methods Transforming i* Goal Models into Business Process Models. In: 2009 17th IEEE International Requirements Engineering Conference. IEEE, pp. 15–23

Dongmo C., Van der Poll J. A. (2014) Addressing the construction of Z and Object-Z with use case maps (UCMs). In: International Journal of Software Engineering and Knowledge Engineering 24(2), pp. 285–327

Dotan D., Rubin J., Yatzkar-Haham T. (2013) Configuration management system for software product line development environment. US Patent 8,549,473

Duran M. B. (2018) Reusable Goal Models. PhD thesis, McGill University, Canada

Duran M. B., Mussbacher G. (2016) Investigation of feature run-time conflicts on goal model-based reuse. In: Information Systems Frontiers, pp. 1–21

Duran M. B., Mussbacher G. (2018) Top-Down Evaluation of Reusable Goal Models. In: New Opportunities for Software Reuse. Springer, pp. 76–92

Duran M. B., Mussbacher G. (2019) Reusability in goal modeling: A systematic literature review. In: Information and Software Technology 110, pp. 156–173

Fan Y., Anda A. A., Amyot D. (2018) An Arithmetic Semantics for GRL Goal Models with Function Generation. In: International Conference on System Analysis and Modeling. Springer, pp. 144–162

Franchitti J.-C. L. (2019) Active adaptation of networked compute devices using vetted reusable software components. US Patent 10,338,913

Genon N., Amyot D., Heymans P. (2011) Analysing the Cognitive Effectiveness of the UCM Visual Notation. In: System Analysis and Modeling: About Models (SAM 2010). Springer, pp. 221–240

Georg G., Mussbacher G., Amyot D., Petriu D., Troup L., Lozano-Fuentes S., France R. (2015) Synergy between Activity Theory and goal/scenario modeling for requirements elicitation, analysis, and evolution. In: Information and Software Technology 59, pp. 109–135

Ghanavati S. (2013) Legal-URN Framework for Legal Compliance of Business Processes. PhD thesis, University of Ottawa, Canada

Ghanavati S., Amyot D., Rifaut A. (2014a) Legal Goal-Oriented Requirement Language (Legal GRL) for Modeling Regulations. In: Proceedings of the 6th International Workshop on Modeling in Software Engineering. MiSE 2014. ACM, pp. 1–6

Ghanavati S., Rifaut A., Dubois E., Amyot D. (2014b) Goal-oriented compliance with multiple regulations. In: 2014 IEEE 22nd International Requirements Engineering Conference (RE). IEEE, pp. 73–82

Ghasemi M., Amyot D. (2019a) Data Preprocessing for Goal-Oriented Process Discovery. In: 27th IEEE International Requirements Engineering Conference Workshops (RE 2019). IEEE, pp. 200–206

Ghasemi M., Amyot D. (2019b) Goal-oriented Process Enhancement and Discovery. In: International conference on business process management. Springer, pp. 102–118

Ghasemi M., Amyot D. (2020) From event logs to goals: a systematic literature review of goal-oriented process mining. In: Requirements Engineering 25(1), pp. 67–93

Gilbreth F. B., Gilbreth L. M. (1921) Process charts. In: Annual Meeting of The American Society of Mechanical Engineers. New York

Godart C., Molli P., Canals G., Lonchamp J. (1992) Extending cooperating transaction model to support software engineering activities. In: Proceedings of the Second International Conference on Systems Integration, pp. 184–192

Gol Mohammadi N. (2019) Systematic Refinement of Trustworthiness Requirements using Goal and Business Process Models In: Trustworthy Cyber-Physical Systems Springer, pp. 119–144

Gonçalves E., Castro J., Araújo J., Heineck T. (2018) A Systematic Literature Review of iStar extensions. In: Journal of Systems and Software 137, pp. 1–33

Guizzardi R., Reis A. N. (2015) A Method to Align Goals and Business Processes. In: Conceptual Modeling. Springer, pp. 79–93

Hashmi M., Governatori G., Lam H., Wynn M. T. (2018) Are we done with business process compliance: state of the art and challenges ahead. In: Knowledge and Information Systems 57(1), pp. 79–133

Hassine J., Amyot D. (2016) A questionnaire-based survey methodology for systematically validating goal-oriented models. In: Requirements Engineering 21(2), pp. 285–308

Hassine J., Amyot D. (2017) An empirical approach toward the resolution of conflicts in goal-oriented models. In: Software & System Modeling 16(1), pp. 279–306

Hassine J., Rilling J., Dssouli R. (2005a) Abstract Operational Semantics for Use Case Maps. In: Formal Techniques for Networked and Distributed Systems - FORTE 2005, 25th IFIP WG 6.1 International Conference. Lecture Notes in Computer Science Vol. 3731. Springer, pp. 366–380

Hassine J., Rilling J., Dssouli R. (2005b) An ASM Operational Semantics for Use Case Maps. In: 13th IEEE International Conference on Requirements Engineering (RE 2005). IEEE, pp. 467–468

Horkoff J., Aydemir F. B., Cardoso E., Li T., Maté A., Paja E., Salnitri M., Piras L., Mylopoulos J., Giorgini P. (2019) Goal-oriented requirements engineering: an extended systematic mapping study. In: Requirements Engineering 24(2), pp. 133–160

Horkoff J., Barone D., Jiang L., Yu E. S. K., Amyot D., Borgida A., Mylopoulos J. (2014) Strategic business modeling: representation and reasoning. In: Software & Systems Modeling 13(3), pp. 1015–1041

Hundal K. S., Mussbacher G. (2018) Model-Based Development with Distributed Cognition. In: 2018 IEEE 8th International Model-Driven Requirements Engineering Workshop (MoDRE). IEEE, pp. 26–35

IBM (2019) IBM ILOG CPLEX Optimization Studio. https://ibm.co/2YRj8oy

IEEE (2012) ISO/IEC/IEEE 31320-1:2012 – Information technology – Modeling Languages–Part 1: Syntax and Semantics for IDEF0. https://www.iso.org/standard/60615.html

Insfrán E., Abrahão S., de Oliveira R. P., González-Ladrón-de-Guevara F., Fernández-Diego M., Cano-Genoves C. (2017) Specifying Value in GRL for Guiding BPMN Activities Prioritization. In: Information Systems Development (ISD 2017)

Intersoft Consulting (2016) General Data Protection Regulation (GDPR). https://gdpr-info.eu/

ISO (2019) ISO/IEC 15909-1:2019 – Systems and software engineering – High-level Petri nets – Part 1: Concepts, definitions and graphical notation. https://www.iso.org/obp/ui/#iso:std:iso-iec:15909:-1:en

ITU-T (2011) Recommendation Z.150 (02/11) User Requirements Notation (URN) - Language requirements and framework. http://www.itu.int/rec/T-REC-Z.150/en

ITU-T (2018) Recommendation Z.151 (10/18) User Requirements Notation (URN) – Language definition. http://www.itu.int/rec/T-REC-Z.151/en

Jacobs S., Holten R. (1995) Goal Driven Business Modelling: Supporting Decision Making within Information Systems Development. In: ACM, pp. 96–105

Karagiannis D., Mayr H. C., Mylopoulos J. (eds.) Domain-Specific Conceptual Modeling, Concepts, Methods and Tools. Springer

Koliadis G., Ghose A. (2006) Relating Business Process Models to Goal-Oriented Requirements Models in KAOS. In: Advances in Knowledge Acquisition and Management (PKAW 2006). Springer, pp. 25–39

Kumar R., Mussbacher G. (2018) Textual User Requirements Notation. In: System Analysis and Modeling (SAM 2018). Springer, pp. 163–182

Kuziemsky C., Liu X., Peyton L. (2010) Leveraging goal models and performance indicators to assess health care information systems. In: Quality of Information and Communications Technology (QUATIC), 2010 Seventh International Conference on the. IEEE, pp. 222–227

Lessard L. (2015a) An Intentional Approach to the Engineering of Knowledge-intensive Service Systems. In: Procedia Manufacturing 3 AHFE 2015, pp. 3383–3390

Lessard L. (2015b) Modeling Value Cocreation Processes and Outcomes in Knowledge-Intensive Business Services Engagements. In: Service Science 7(3), pp. 181–195

Lessard L., Amyot D., Aswad O., Mouttham A. (2020) Expanding the nature and scope of requirements for service systems through Service-Dominant Logic: the case of a telemonitoring service. In: Requirements Engineering 25(3), pp. 273–293

Liaskos S., Jalman R., Aranda J. (2012) On eliciting contribution measures in goal models. In: 2012 20th IEEE International Requirements Engineering Conference (RE). IEEE, pp. 221–230

Liscano R., Fullarton S., Mussbacher G., Gray T. (2008) System and method for remote assembly of messages to create a control message. US Patent 7,318,109

Liu L., Yu E. (2004) Designing information systems in social context: a goal and scenario modelling approach. In: Information Systems 29(2), pp. 187–203

Liu Y., Su Y., Yin X., Mussbacher G. (2014) Combined propagation-based reasoning with goal and feature models. In: 2014 IEEE 4th International Model-Driven Requirements Engineering Workshop (MoDRE). IEEE, p. 2736

Lo A., Yu E. S. K. (2007) From Business Models to Service-Oriented Design: A Reference Catalog Approach. In: Conceptual Modeling (ER 2007). Lecture Notes in Computer Science Vol. 4801. Springer, pp. 87–101

Massey A., Holtgrefe E., Ghanavati S. (2017) Modeling Regulatory Ambiguities for Requirements Analysis. In: Conceptual Modeling (ER2017, pp. 231–238

Massey A., Rutledge R., Antón A., Swire P. (2014) Identifying and Classifying Ambiguity for Regulatory Requirements. In: 2014 IEEE 22nd International Requirements Engineering Conference (RE). IEEE, pp. 83–92

Mears M. G., Nierzwick M., Pash P. E., Rizzo V. R., Steiner B., Westerfield K. M. (2016) Handheld diabetes manager with a user interface for displaying a status of an external medical device. US Patent 9,252,870

Mili H., Tremblay G., Jaoude G. B., Lefebvre É., Elabed L., Boussaidi G. E. (2010) Business Process Modeling Languages: Sorting through the Alphabet Soup. In: ACM Computing Surveys 43(1)

Mokhati F., Menassel Y. (2013) Towards formalising use case maps in Maude strategy language: application to multi-agent systems. In: International journal of computer applications in technology 47(2-3), pp. 138–151

Moody D. L., Heymans P., Matulevicius R. (2010) Visual syntax does matter: improving the cognitive effectiveness of the *i** visual notation. In: Requirements Engineering 15(2), pp. 141–175

Mussbacher G., Amyot D. (2008) Assessing the Applicability of Use Case Maps for Business Process and Workflow Description. In: 2008 International MCETECH Conference on e-Technologies (MCETECH 2008). IEEE, pp. 219–222

Mussbacher G. (2010) Aspect-oriented User Requirements Notation. PhD thesis, University of Ottawa, Canada

Mussbacher G., Amyot D., Araújo J., Moreira A. (2010a) Requirements Modeling with the Aspect-oriented User Requirements Notation (AoURN): A Case Study In: Transactions on Aspect-Oriented Software Development VII: A Common Case Study for Aspect-Oriented Modeling Springer, pp. 23–68

Mussbacher G., Amyot D., Whittle J. (2009) Refactoring-Safe Modeling of Aspect-Oriented Scenarios. In: Model Driven Engineering Languages and Systems. Springer, pp. 286–300

Mussbacher G., Amyot D., Whittle J. (2013) Composing Goal and Scenario Models with the Aspect-Oriented User Requirements Notation Based on Syntax and Semantics In: Aspect-Oriented Requirements Engineering Springer, pp. 77–99

Mussbacher G., Araújo J., Moreira A., Amyot D. (2012) AoURN-based modeling and analysis of software product lines. In: Software Quality Journal 20(3–4), pp. 645–687 10.1007/s11219-011-9153-8

Mussbacher G., Whittle J., Amyot D. (2010b) Modeling and detecting semantic-based interactions in aspect-oriented scenarios. In: Requirements Engineering Journal (REJ) 15(2), pp. 197–214

Mylopoulos J., Borgida A., Jarke M., Koubarakis M. (1990) Telos: Representing Knowledge about Information Systems. In: ACM Transactions on Information Systems 8(4), pp. 325–362

Mylopoulos J., Chung L., Nixon B. A. (1992) Representing and using nonfunctional requirements: a process-oriented approach. In: IEEE Transactions on Software Engineering 18(6), pp. 483–497

Ni Q., Bertino E., Lobo J., Brodie C., Karat C., Karat J., Trombetta A. (2010) Privacy-aware role-based access control. In: ACM Transactions on Information and System Security 13(3), 24:1–24:31

Odeh Y., Green S., Odeh M. (2018) Deriving Goal-Oriented Models from Business Process Models: Applied to Cancer Care Organisation. In: 2018 1st International Conference on Cancer Care Informatics (CCI). IEEE, pp. 125–135

OMG (2014) Business Process Model and Notation (BPMN), version 2.0.2. https://www.omg.org/spec/BPMN/2.0.2

OMG (2017) Unified Modeling Language (UML), version 2.5.1. https://www.omg.org/spec/UML/2.5.1

Patrício L., de Pinho N. F., Teixeira J. G., Fisk R. P. (2018) Service Design for Value Networks: Enabling Value Cocreation Interactions in Healthcare. In: Service Science 10(1), pp. 76–97

Perera H., Mussbacher G., Hussain W., Ara Shams R., Nurwidyantoro A., Whittle J. (2020) Continual Human Value Analysis in Software Development: A Goal Model Based Approach. In: 2020 IEEE 28th International Requirements Engineering Conference (RE), pp. 192–203

Pesic M., Van der Aalst W. M. (2006) A declarative approach for flexible business processes management. In: International conference on business process management. Springer, pp. 169–180

Petriu D. B., Amyot D., Woodside C. M. (2003) Scenario-Based Performance Engineering with UCMNAV. In: System Design, 11th International SDL Forum (SDL 2003). Lecture Notes in Computer Science Vol. 2708. Springer, pp. 18–35

Pinciroli F., Barros Justo J. L., Forradellas R. (2020) Systematic mapping study: On the coverage of aspect-oriented methodologies for the early phases of the software development life cycle. In: Journal of King Saud University - Computer and Information Sciences

Rahman A., Amyot D. (2014) A DSL for importing models in a requirements management system. In: IEEE 4th International Model-Driven Requirements Engineering Workshop (MoDRE 2014). IEEE, pp. 37–46

Roy J.-F., Kealey J., Amyot D. (2006) Towards Integrated Tool Support for the User Requirements Notation. In: System Analysis and Modeling: Language Profiles. Springer, pp. 198–215

Sartoli S., Ghanavati S., Siami Namin A. (2020a) Compliance Requirements Checking in Variable Environments. In: 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC). IEEE, pp. 1093–1094

Sartoli S., Ghanavati S., Siami Namin A. (2020b) Towards Variability-Aware Legal-GRL Framework for Modeling Compliance Requirements. In: 2020 IEEE 7th International Workshop on Evolving Security Privacy Requirements Engineering (ESPRE). IEEE, pp. 7–12

Shamsaei A. (2012) Indicator-based policy compliance of business processes. PhD thesis, University of Ottawa, Canada

Shamsaei A., Pourshahid A., Amyot D. (2010) Business Process Compliance Tracking Using Key Performance Indicators. In: Business Process Management Workshops (BPM 2010). Lecture Notes in Business Information Processing Vol. 66. Springer, pp. 73–84

Sousa H. P., Leite J. C. S. d. P. (2014) Modeling Organizational Alignment. In: Conceptual Modeling - 33rd International Conference (ER2014). Lecture Notes in Computer Science Vol. 8824. Springer, pp. 407–414

U.S. Government (1996) Health Insurance Portability and Accountability Act (HIPAA). http://www.hhs.gov/ocr/privacy/

Van Der Aalst W. (2016) Data science in action. In: Process mining. Springer, pp. 3–23

van Lamsweerde A. (2001) Goal-oriented requirements engineering: a guided tour. In: Proceedings Fifth IEEE International Symposium on Requirements Engineering. IEEE, pp. 249–262

van Lamsweerde A. (2009) Requirements engineering: From system goals to UML models to software Vol. 10. Chichester, UK: John Wiley & Sons

Vigder M. (1992) Applying formal techniques to the design of concurrent systems.. PhD thesis, Carleton University, Canada

Vinay S., Aithal S., Sudhakara G. (2014) Effect of Contribution Links on Choosing Hard Goals in GORE Using AHP and TOPSIS. In: Emerging Research in Electronics, Computer Science and Technology: Proceedings of International Conference, ICERECT 2012. Springer, pp. 737–745

Vrbaski M., Mussbacher G., Petriu D. C., Amyot D. (2012) Goal models as run-time entities in context-aware systems. In: 7th Workshop on Models@run.time. ACM, pp. 3–8

Weigert T., Kolchin A., Potiyenko S., Gurenko O., van den Berg A., Banas V., Chetvertak R., Yagodka R., Volkov V. (2019) Generating Test Suites to Validate Legacy Systems. In: System Analysis and Modeling, 11th International Conference (SAM 2019). Lecture Notes in Computer Science Vol. 11753. Springer, pp. 3–23

Weigert T. J. (2017) System and method for iterative generating and testing of application code. US Patent 9,778,916

Yu E., Amyot D., Mussbacher G., Franch X., Castro J. (2013) Practical applications of *i** in industry: The state of the art. In: Requirements Engineering Conference (RE), 2013 21st IEEE International. IEEE, pp. 366–367

Yu E. S. K. (1997) Towards modelling and reasoning support for early-phase requirements engineering. In: 3rd IEEE International Symposium on Requirements Engineering (ISRE'97). IEEE, pp. 226–235

Yu E. S., Giorgini P., Maiden N., Mylopoulos J. (2011) Social modeling for requirements engineering. MIT press