Consistency Analysis for User Requirements Notation Models

Okhaide Akhigbe[1], Daniel Amyot[1], Amal Ahmed Anda[1], Lysanne Lessard[2], and Daoyang Xiao[2]

[1] School of Electrical Engineering and Computer Science,
[2] Telfer School of Management
University of Ottawa, Ottawa, Canada
{okhaide, damyot, aanda027, dxiao089}@uottawa.ca,
lessard@telfer.uottawa.ca

Abstract. The User Requirements Notation (URN) is a standard modeling language that includes two complementary views, one for goals with the Goal-oriented Requirement Language (GRL) and one for scenarios/processes with Use Case Maps (UCM). The URN standard, however, does not provide means of checking consistency between the GRL and UCM views, leading to models that are potentially erroneous. This paper presents a preliminary set of rules for checking common consistency properties in URN models. These rules have been implemented as user-selectable OCL constraints in the jUCMNav tool. Future opportunity for research in that space are also identified.

Keywords: User Requirements Notation · GRL · Goal · Scenario · Consistency · OCL · jUCMNav

1 Introduction

The *User Requirements Notation* (URN) is a standard modeling language used to model and analyze requirements [1, 2]. URN includes two complementary views, the *Goal-oriented Requirement Language* (GRL) for modeling goals, actors, indicators, and their relationships, and the *Use Case Map* (UCM) notation for modeling responsibilities, processes/scenarios, and their underlying components. GRL and UCM models can be connected using typed *URN Links* by specifying user-defined relationships between any pair of URN model elements. URN Links establish traceability and enable completeness and consistency analysis between GRL and UCM models.

Although the URN standard provides well-formedness rules to ensure some consistency within each of the GRL and UCM views, it does not provide means of checking consistency *across* these views, leading to models that are potentially erroneous.

To address this gap, we present a preliminary set of user-selectable rules for checking common consistency properties in URN models in a way that exploits URN Links as well as URN metadata (i.e., annotations on URN model elements). The rules were created using UML's *Object Constraint Language* (OCL) [3], and implemented in the jUCMNav tool. jUCMNav is a free plugin for Eclipse used to create and analyze

URN models [4]. This paper is a first step towards user-selectable rules supporting automated consistency analysis between the GRL and UCM views of a URN model.

The remainder of the paper is as follows. Section 2 describes work related to consistency analysis in modeling languages. A modeling example is used to describe consistency problems in URN models in Section 3, while Section 4 highlights our proposed consistency rules. This paper concludes in Section 5 with a summary and a list of potential consistency issues we hope to address in the future.

2 Related Work

Consistency analysis in modeling aims to detect contradictions when multiple views are used to specify different subsets of a model. Inconsistencies occur frequently if these views are provided by different modelers, or when a language includes different sub-notations. A good example is UML, where several challenges exist due to the presence of 14 diagram types. For instance, messages used in sequence diagrams and transition actions used in state machine diagrams must correspond to operations (and their signatures) in their respective classes in class diagrams. There is even a workshop focusing on such issues [5]. Resolving consistency issues often involves adding, deleting, or modifying model elements in one or many views to better align the latter with each other.

Goal models have also been connected to other types of models, with consistency challenges. Alves et al. [6] have investigated bi-directional mappings between *Business Process Model and Notation* (BPMN) and *i** models. Their mappings targeted transformations (instead of checking), but these mappings were not defined in a way amenable to automation. A similar informal and syntactic approach was explored by Guizzardi and Reis for BPMN and Tropos models [7]. In their GoalBPM approach, Koliadis and Ghose trace BPMN elements to KAOS goals through effect annotations [8]. Sousa and Leite proposed to merge BPMN, *i** and indicators into a single language in their GPI approach, in order to simplify the alignment between goals and processes. However, they do not suggest consistency or completeness rules.

Our work is inspired from these related contributions, but also targets automated checking of inconsistencies in a standard multi-view language (URN) through user-selectable rules formalized in OCL.

3 Consistency Issues in URN Models

Consider the illustrative example described in Fig. 1. In the GRL view shown on the left, we have a Transportation System whose At Work goal is decomposed into two options described as tasks: By Bus or By Bike. The Worker actor wants to improve her health, and the first option hurts that softgoal whereas the second option helps it. The top UCM map shown in Fig. 1 describes the process involved in the worker getting to work. The choice of transportation mode is described using a dynamic stub (Transport) and two options are available, TakeCar and TakeBus, captured as submaps with one responsibility each (and plugged into that stub).

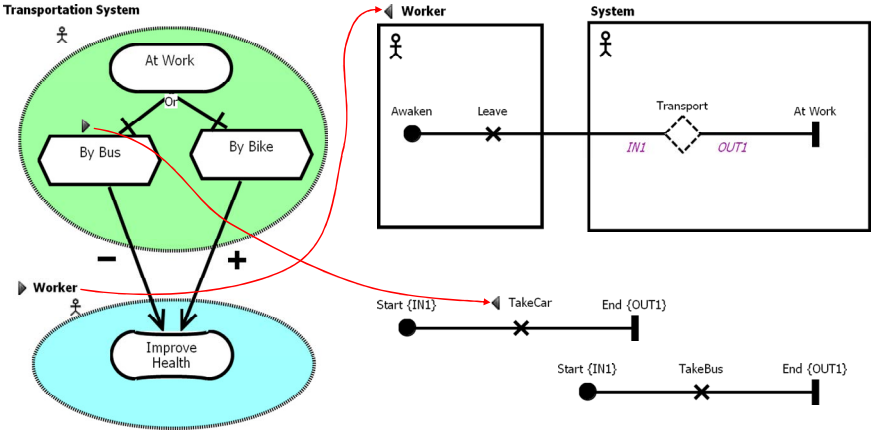

Fig. 1. Sample model showing one GRL diagram and three UCM diagrams

URN Links of type *Traces* are used to capture explicit traceability information used during consistency analysis, and links are visualized through outgoing (▸) and incoming (◂) link triangles. In this example, the Worker actor in the GRL view traces to the Worker component in the UCM view. In addition, the By Bus task in the GRL view traces (incorrectly) to the TakeCar responsibility in the UCM view. The connecting arrows were added for illustration but they are not part of the URN notation.

We can observe potential inconsistencies across these GRL and UCM views. The GRL Transportation System should likely trace to the UCM System. The By Bike task has no corresponding map or responsibility, which can raise several questions (is this task spurious? Is a map or responsibility missing?). The By Bus task likely traces to the incorrect responsibility (should be to TakeBus instead of TakeCar), but then the TakeCar map might require a link from a new GRL task or be removed for consistency. Lastly, the Leave responsibility has no trace link, but maybe there is no reason to link it either; not all fine-grained operationalizations need to be justified, just like not all goals need to be operationalized (e.g., the color of the system shall be blue).

Given the current lack of support for consistency analysis in the URN standard, these types of inconsistencies (and others) could easily be overlooked in more complex models, leading to erroneous model analysis results. The resolution of such inconsistencies (e.g., adding/removing/modifying a GRL/UCM model element or URN Link, or tagging a model element as something that does not require traceability) requires human judgement and is outside the scope of this work. However, consistency rules can still be defined formally in order to detect violations automatically.

4 Goal-Process Consistency Rules for URN Models

Consistency rules can be defined as constraints on a language's metamodel. OCL is a good choice in our context as URN has a unified metamodel covering both the GRL and UCM views. From an implementation perspective, jUCMNav supports the defini-

tion, selection, and checking of OCL rules by modelers, and a library of predefined OCL functions for extracting information from URN models is also provided [10]. Such mechanisms enable the checking of rules, e.g., in a language profile [11].

Our consistency rules exploit two modeler-provided information elements: URN Links of type *Traces* from a GRL element to a UCM element (by convention), and metadata (name=Traces, value=No) used to annotate model elements for which we do not expect any consistency-related *Traces* link. Rules come in groups of multiple, fine-grained OCL invariants in order to report violations with precise error messages.

The first rule (#1, below) addresses the consistency between GRL actors and UCM components. Part #1a checks the existence of a *Traces* link sourced at the GRL actor and part #1b checks that the target of such links is indeed a UCM component. These invariants make use of OCL functions predefined in our library: invoking `getMetadata(n:String):String` returns the value of the first attached metadata object whose name is `n`, and `getLinksToForType(t:String):Sequence(URNmodelElement)` returns a collection of target model elements found at the end of URN Links of type `t`.

```
context grl::Actor
inv URNconsAllActorsToComponents:
   -- #1a: Each GRL actor must have a Traces link to a UCM component,
   -- unless tagged with Traces=No
   not(getMetadata('Traces')='No') implies
         (getLinksToForType('Traces')->size() > 0)

inv URNconsActorsToComponentsOnly:
   -- #1b: Traces links from a GRL actor must only be to a UCM *component*
   not(getMetadata('Traces')='No') implies
         ( getLinksToForType('Traces')->
           forAll(me:urncore::URNmodelElement |
                 me.oclIsKindOf(urncore::Component) ) )
```

The above invariants check this rule from the actors' perspective only. However, one must also check the presence of links from the UCM components' perspective (as there could be components without incoming *Traces* links). Hence, another pair of similar invariants is provided (#1c and #1d in Table 1). A violation of any of these invariants reported by jUCMNav on a URN model will indicate which model element is incorrect, and the description is returned as an error message. For the example in Fig. 1, jUCMNav reports violations of invariant #1a on the GRL actor Transportation System and of invariant #1c on the UCM component System.

The second rule in Table 1, composed of six invariants, checks the consistency between GRL intentional elements and UCM maps *or* responsibilities (in order to cover different levels of granularity). For example, in Fig. 1, the GRL goal At Work could have a *Traces* link to the top-level UCM map, and the Leave responsibility could be tagged with a *Traces=No* metadata. Rule 3 is an alternative to Rule 2 where only GRL tasks are used for consistency checking instead of any type of GRL intentional element (assuming that covering tasks is enough in some modeling context). The modeler would need to choose between Rule 2 and Rule 3 for the consistency analysis as these rules are mutually exclusive.

Table 1. Identifiers and descriptions of OCL consistency rules for URN models

ID	Description
1a	Each GRL actor must have a Traces link to a UCM component, unless tagged with Traces=No
1b	Traces links from a GRL actor must only be to a UCM *component*
1c	Each UCM component must have a Traces link from a GRL actor, unless tagged with Traces=No
1d	Traces links to a UCM component must only be from a GRL *actor*
2a	Each GRL intentional element must have a Traces link to a UCM map or responsibility, unless tagged with Traces=No
2b	Traces links from a GRL intentional element must only be to a UCM *map or responsibility*
2c	Each UCM map must have a Traces link from a GRL intentional element, unless tagged with Traces=No
2d	Traces links to a UCM map must only be from a GRL *intentional element*
2e	Each UCM responsibility must have a Traces link from a GRL intentional element, unless tagged with Traces=No
2f	Traces links to a UCM responsibility must only be from a GRL *intentional element*
3a	Each GRL task must have a Traces link to a UCM map or responsibility, unless tagged with Traces=No
3b	Traces links from a GRL task must only be to a UCM *map or responsibility*
3c	Each UCM map must have a Traces link from a GRL task, unless tagged with Traces=No
3d	Traces links to a UCM map must only be from a GRL *task*
3e	Each UCM responsibility must have a Traces link from a GRL task, unless tagged with Traces=No
3f	Traces links to a UCM responsibility must only be from a GRL *task*
4c	Each UCM component (or one of its parents) must have a Traces link from a GRL actor, unless tagged with Traces=No
4d	Traces links to a UCM component (or one of its parents) must only be from a GRL *actor*

Rule 4 is a recursive version of Rule 1 where #4a=#1a and #4b=#1b. From a UCM perspective, it suffices that the component or one of its containing components is the target of a *Traces* link from a GRL actor for the rule to be satisfied. For example, if component C2 is the target of a valid Traces link and C1 contains C2 and C2 contains C3, then C2 and C3 would be consistent (no violation) whereas C1 would not. Other such recursive rules, which minimize the need to manually create *Traces* URN links, could also be considered (e.g., for a structure of maps with stub containing sub-maps).

5 Conclusions and Future Work

In this paper, using an example, we showed a preliminary set of consistency rules for automatically checking common consistency properties across the goal and process views in URN models. The 18 invariants in Table 1 have been implemented as user-selectable OCL constraints in the jUCMNav tool, and tested for correctness. Other such rules are currently under development and are expected to be released as a catalogue of validated consistency rules, possibly integrated to the URN standard.

This work raises many new research questions on how best to achieve sound goal-process consistency in URN models while maximizing usability through automation (as manually creating and checking URN links is much cumbersome). For example:

- Should there be different types of links connecting GRL and UCM elements?
- Can UCM variables (used in conditions and responsibilities) capturing the satisfaction levels of GRL intentional elements be used as traceability links?
- What recursive rules exploiting containment structures are beneficial?
- Should different rules apply to different parts/elements of a URN model?
- Can natural language processing (NLP) or other approaches be used for creating (some) URN links automatically?
- How can jUCMNav's interface be made more usable? E.g., when we create a GRL actor, should a UCM component be created and linked automatically?
- In addition to the "syntactic" rules proposed in this paper, should semantic rules and evolution rules (consistency across versions) be considered [5]?

An empirical study on the application of such rules would also help observe what really helps or hurts in terms of consistency and overall quality of URN models.

Acknowledgments. OA and DX are supported by Discovery grants (of DA ad LL respectively) from the Natural Science and Engineering Research Council of Canada. OA is further supported by Interis Consulting/BDO, and AAA by a scholarship from the Government of Libya.

References

1. International Telecommunication Union: Recommendation Z.151(10/12), User Requirements Notation (URN) – Language Definition. Geneva, Switzerland, 2012.
2. Amyot, D., Mussbacher, G.: User Requirements Notation: The First Ten Years, The Next Ten Years. Journal of Software (JSW), Vol. 6, No. 5, 747–768, 2011.
3. Object Management Group: Object Constraint Language (OCL), Version 2.4, February 2014.
4. Amyot, D., Shamsaei, A., Kealey, J., Tremblay E., Miga, A., Mussbacher, G., Tawhid, R., Braun, E., Catwright, N.: Towards Advanced Goal Model Analysis with jUCMNav, in ER Workshops 2012, Springer, pp. 201–210, 2012.
5. Torre, D., Labiche, Y., Genero, M., Elaasar, M., Das, T.K., Hoisl, B., Kowal, M.: WUCOR 2015: Post workshop report. ACM SIGSOFT SEN, 41(2): 34–37 (2016)
6. Alves, R., Silva, C., Castro, J.: A bi-directional mapping between i* and BPMN models in the context of business process management. ER@BR 2013. CEUR-WS Vol-1005 (2013)
7. Guizzardi, R., Reis, A.N.: A Method to Align Goals and Business Processes. Conceptual Modeling. LNCS 9381, Springer, 79–93 (2015)
8. Koliadis, G., Ghose, A.: Relating Business Process Models to Goal-Oriented Requirements Models in KAOS. PKAW 2006. NCS 4303, Springer, 25–39 (2006)
9. Sousa, H.P., Leite, J.C.S.P: Modeling Organizational Alignment. Conceptual Modeling. LNCS 8824, 407–414 (2014)
10. Amyot, D., Yan, J.B.: Flexible Verification of User-Defined Semantic Constraints in Modelling Tools. CASCON 2008. ACM Press, 81–95 (2008)
11. Amyot, D., Horkoff, J., Gross, D., Mussbacher, G.: A Lightweight GRL Profile for i* Modeling. RIGiM 2009, ER Workshops, LNCS 5833. Springer, 254–264 (2009)