Combined Goal and Feature Model Reasoning with the User Requirements Notation and jUCMNav

Yanji Liu, Yukun Su, Xinshang Yin, Gunter Mussbacher
ECE, McGill University
Montréal, Canada
{yanji.liu | yu.k.su | xinshang.yin}@mail.mcgill.ca, gunter.mussbacher@mcgill.ca

Abstract— **The User Requirements Notation (URN) is an international requirements engineering standard published by the International Telecommunication Union. URN supports goal-oriented and scenario-based modeling and analysis. jUCMNav is an open-source, Eclipse-based modeling tool for URN. This tool demonstration focuses on recent extensions to jUCMNav that have incorporated feature models into a URN-based modeling and reasoning framework. Feature modeling is a well-establishing technique for capturing commonalities and variabilities of Software Product Lines. Combined with URN, it is possible to reason about the impact of feature configurations on stakeholder goals and system qualities, thus helping to identify the most appropriate features for a stakeholder. Furthermore, coordinated feature and goal model reasoning is fundamental to Concern-Driven Development, where concerns are defined with a three-part variation, customization, and usage interface. As the variation interface is described with feature and goal models, it is now possible with jUCMNav to define and reason about a concern's variation interface, which is a prerequisite for composing multiple concerns based on their three-part interfaces.**

Index Terms—**goal modeling, feature modeling, GRL, Goal-oriented Requirement Language, URN, User Requirements Notation, trade-off analysis, evaluation algorithm, concern, concern-driven development, jUCMNav.**

I. Introduction

The User Requirements Notation (URN) is an international requirements engineering (RE) standard published by the International Telecommunication Union in the Z.15x series [5]. URN targets the modeling and analysis of goals and scenarios, by combining the Goal-oriented Requirement Language (GRL) with the Use Case Map (UCM) notation. For more than a decade [2], URN has proven and continues to be useful and successful for many RE activities, in academia and industry. The free, open-source, Eclipse-based jUCMNav tool has contributed significantly to this success, as it allows URN models to be defined, navigated, analyzed, and transformed [6].

Recently, it was suggested to combine goal-oriented and feature-based modeling for more sophisticated reasoning [7] in the context of Software Product Lines (SPLs) [8] and Concern-Driven Development (CDD) [1]. SPLs are a structured reuse approach and have traditionally used feature models to identify the commonalities and variabilities of a family of similar products. The objective of Concern-Driven Development (CDD) is to provide next generation reuse technologies, through a three-part interface describing units of reuse, i.e., concerns. The variation interface describes required design decisions and their impact on high level system qualities with the help of feature and goal models. The customization interface allows the chosen variation to be adapted to a specific reuse context, while the usage interface defines how a customized concern may eventually be used.

The next section introduces three contributions to facilitate combined reasoning of feature and goal models in jUCMNav: (i) support for the definition of feature models based on goal model semantics, (ii) simultaneous analysis of feature and goal models through a single, shared evaluation algorithm, and (iii) improved analysis based on feature model semantics capable of identifying invalid feature configurations and automatically selecting mandatory features.

II. Feature Models and URN

To support the modeling and analysis of feature models in jUCMNav, the metamodel for GRL was extended with feature modeling concepts by subclassing existing GRL metaclasses. A *feature* (⬡, e.g., Energy Management in Fig. 1) is a subclass of GRL intentional elements (i.e., the nodes in a GRL model). A ⬡ is chosen as features are conceptually close to GRL tasks (also shown with ⬡), while a ☐ represents the very different concept of GRL resources. *Optional links* (—o, e.g., between Energy Management and Air Conditioning Control) and *mandatory links* (—●, e.g., between Energy Management and Light Control) are subclasses of GRL contribution links. Since feature models are an and/or tree and goal models are an and/or graph, *alternative feature groups* and *OR feature groups* can be modeled with existing XOR and OR decomposition links in GRL, respectively (note that AND decomposition links are an alternative way of modeling mandatory links). Cross-tree integrity constraints such as "requires" or "conflicts" can be defined with OCL constraints [7], which have been supported by jUCMNav since version 3.1. Finally, a *feature configuration* is a subclass of the GRL strategy concept. Both allow a set of model elements to be selected for analysis purposes.

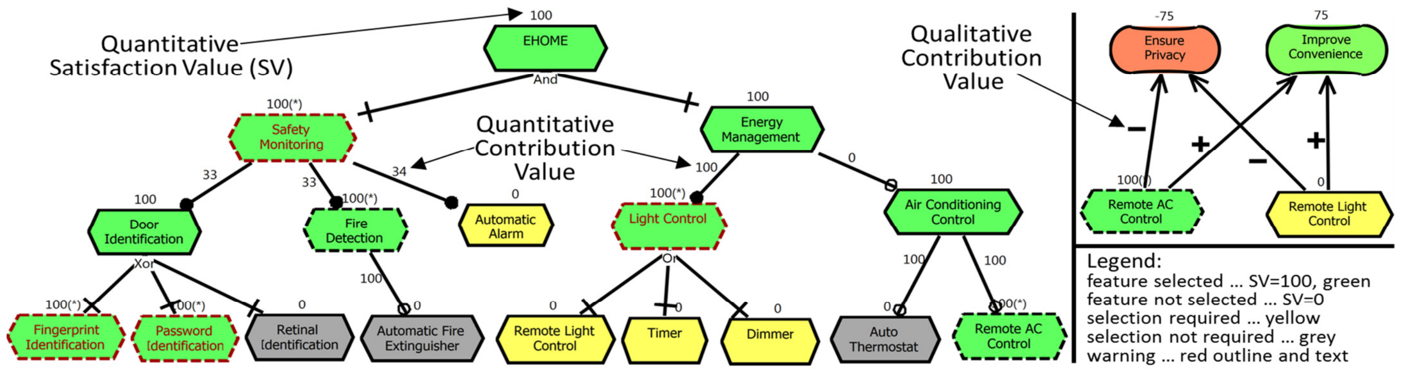

Fig. 1. Feature Modeling and Analysis in jUCMNav

In GRL, each selection can be quantified with a satisfaction value between -100 (element is not satisfied at all) and 100 (fully satisfied), while in the context of feature models, the selection is constrained to 0 (for not part of the feature configuration) and 100 (for part of the feature configuration).

The advantage of this approach is that (i) the infrastructure for evaluation algorithms [3] is readily available for feature models and (ii) features can be used in GRL models to define the impact of features on stakeholder goals and system qualities, thus enabling combined reasoning about feature and goal models for stakeholder trade-offs. The single evaluation algorithm shared between feature and goal models only needs to understand how to interpret optional and mandatory links, because features/intentional elements, decomposition links/feature groups, feature configuration/strategies, and integrity/OCL constraints are semantically equivalent in terms of their required behavior for the evaluation algorithm. This new evaluation algorithm is called the "Feature Model Strategy Algorithm" and can be selected in the preference settings of jUCMNav. The result of a feature model evaluation with the customary color-coded feedback is shown on the left side of Fig. 1. Note that standard GRL goal models capture the impact of features on goals (e.g., Remote Light Control and Remote AC Control have a negative impact on Ensure Privacy but a positive impact on Improve Convenience) as shown briefly on the right.

In addition, several warnings have been implemented that indicate with a red dashed outline/text to the modeler where in the feature model a feature configuration is currently invalid (in Fig. 1, e.g., a mandatory child of Safety Monitoring is not selected, at least one child of Light Control is not selected, and at the most one child of Door Identification can be selected). Optionally, auto-selection of mandatory features can be activated via jUCMNav's preferences. If activated, all mandatory and AND-decomposed children of a feature are automatically selected, when evaluating the feature model.

III. Conclusions and Future Work

This paper gives an overview of three recent extensions to jUCMNav, the currently most comprehensive URN modeling tool, for the purpose of integrated feature and goal model reasoning. This enables jUCMNav to be used in the context of SPL development, in general, and for CDD in particular. In the context of goal modeling for SPLs, several approaches [4][9][10] offer only some but not all of (i) feature modeling, (ii) the modeling of impact of features on goals, (iii) reasoning about stakeholder trade-offs, and (iv) tool support in contrast to URN/jUCMNav. In the context of CDD, jUCMNav is currently the only tool that supports requirements engineering activities for CDD. In future work, we plan to formalize the composition of concerns based on the three-part interface and support such composition in the jUCMNav tool.

References

[1] Alam, O., Kienzle, J., and Mussbacher, G., "Concern-Oriented Software Design", Model Driven Engineering Languages and Systems, Springer, 2013, LNCS 8107:604–621. DOI: 10.1007/978-3-642-41533-3_37.

[2] Amyot, D. and Mussbacher, G., "User Requirements Notation: The First Ten Years, The Next Ten Years", Journal of Software (JSW), 2011, 6(5):747–768. DOI: 10.4304/jsw.6.5.747-768.

[3] Amyot, D., Ghanavati, S., Horkoff, J., Mussbacher, G., Peyton, L., and Yu, E., "Evaluating Goal Models within the Goal-oriented Requirement Language", IJIS, Wiley, 2010, 25(8):841–877. DOI: 10.1002/int.20433.

[4] Bagheri, E., Noia, T., Ragone, A., and Gasevic, D., "Configuring Software Product Line Feature Models Based on Stakeholders' Soft and Hard Requirements", Software Product Lines: Going Beyond, Springer, 2010, LNCS 6287:16–31. DOI: 10.1007/978-3-642-15579-6_2.

[5] International Telecommunication Union, "Recommendation Z.151 (10/12), User Requirements Notation (URN) – Language definition", 2012. http://www.itu.int/rec/T-REC-Z.151/en

[6] jUCMNav website, version 6.0, University of Ottawa. http://jucmnav.softwareengineering.ca/jucmnav

[7] Mussbacher, G., Araújo, J., Moreira, A., and Amyot, D., "AoURN-based Modeling and Analysis of Software Product Lines", Software Quality Journal (SQJ), Springer, 2011, 20(3-4):645–687. DOI: 10.1007/s11219-011-9153-8.

[8] Pohl, K., Böckle, G., and van der Linden, F.J., "Software Product Line Engineering: Foundations, Principles and Techniques", Springer, 2005.

[9] Than Tun, T., Boucher, Q., Classen, A., Hubaux, A., and Heymans, P., "Relating Requirements and Feature Configurations: A Systematic Approach", 13th Intl. Software Product Line Conference (SPLC'09), CMU, 2009, 201–210.

[10] Yu, Y., Leite, J., Lapouchnian, A., and Mylopoulos, J., "Configuring Features with Stakeholder Goals", SAC'08, ACM, 2008, 645–649. DOI: 10.1145/1363686.1363840.