Advanced Science and Technology Letters
Vol.28 (NT 2013), pp.19-24
http://dx.doi.org/10.14257/astl.2013.28.05

Constructing Domain Feature Models Based on the i* Framework

Yifei Zhang[1], Dongjin Yu[1] and Wei Wu[2]

[1] School of Computer, Hangzhou Dianzi University, Hangzhou, China
[2] Zhejiang Provincial Key Laboratory of Network Technology and Information Security, Hangzhou, China
yudj@hdu.edu.cn, ww@topcheer.cn

Abstract. In recent years, a range of approaches that map the goal models to the feature models are proposed in order to construct the traceability of commonality and variability during the Software Product Line Engineering. However, these approaches can only map part of elements. How to map the dependencies among actors to feature models is still left as an open problem. In this paper, we adopt the standard goal-oriented requirements engineering framework, i.e., the i* framework, for constructing the domain feature model. We propose the mapping rules and also the algorithm of mapping dependencies among actors to feature models. The experiment shows that our approach is practical.

Keywords: goal-oriented approach; i*framework; software product line; feature model; application product customization

1 Introduction

In the process of software development based on Software Product Line (SPL), the feature, as a cohesive set of individual requirements, is used to describe the domain commonality and variability [1] [2]. Although the feature expresses a user capability of a software system, it cannot describe non-system (environment) factor and quality characteristic [3]. Moreover, since the feature model is just a concise taxonomic form, it cannot depict the “why” problem, i.e., why a system would interact with other actors, why an actor would engage in some processes in a certain way.

Different with the feature modeling, the goal-oriented approaches describe system states and behavior, i.e., answer not only the ”what” and ”how” questions, and also the ”why” questions. Currently, the goal-oriented modeling and its common approaches such as the i* framework [4][5], KAOS and GBRAM are widely studied, among which the i* framework is a well-known comprehensive approach describing goal and goal dependency.

Transforming the goal models to the feature models expresses a process from acquiring the domain requirement to analyzing it, thus explains both the source and the expression of requirements. In recent years, many approaches of mapping features from goals to trace commonality and variability are proposed in SPL. However, these

ISSN: 2287-1233 ASTL

approaches can only map part of elements. To map the dependencies among actors to the feature models is still left as an open problem. In addition, there is lack of essential associations among the feature models which are constructed by different actors. In this paper, we adopt the i* framework to construct the domain feature model. We propose the rules and the algorithm used to map dependencies among the actors to the more complete feature models. Besides, we initiate to customize the application products using the i* model evaluation approach [6] [7] during the application engineering phase. By using our approach, the reason of selecting the specific features to customize an application product can be clearly elucidated.

The rest of the paper is organized as follows. After Section 2 presents the mapping rules between the i* framework and the feature model, Section 3 describes in detail the whole process to construct the domain feature model based on the i* framework. The case studies are given in Section 4. Finally, the last section concludes the paper and outlines the future work.

2 Rules Mapping between the i* Framework and the Feature Model

Diverse combinations of the goal models and feature models are shown in different structural types where some corresponding traceability may exist. We conclude 7 rules mapping between the i* framework and the feature models, which are shown in Fig. 1.

3 Constructing the Domain Feature Model Based on the i* Framework

In this section, we provide an automatic algorithm of constructing initial domain feature model based on the i*framework, which could be then adjusted to form the final domain feature model.

3.1 Constructing the initial domain feature model

The i* framework may contain several actors that are expressed as types of software system and environment, each of them interacting with others to achieve the goals. First of all, we should sequentially select the actor expressing the software system as the mapping object because features cannot depict non-system (environment) factor.

Based on [8], we propose the algorithm for constructing the feature tree model by mapping dependencies among actors. Firstly, we select an actor randomly and generate a collection that contains all the elements which possess dependencies. Then, by applying the rules given above, we map the dependency elements and links to the feature fragments in order to associate features belonging to different feature trees. Thus, an initial domain feature model can be formed after all actors are treated in this way.

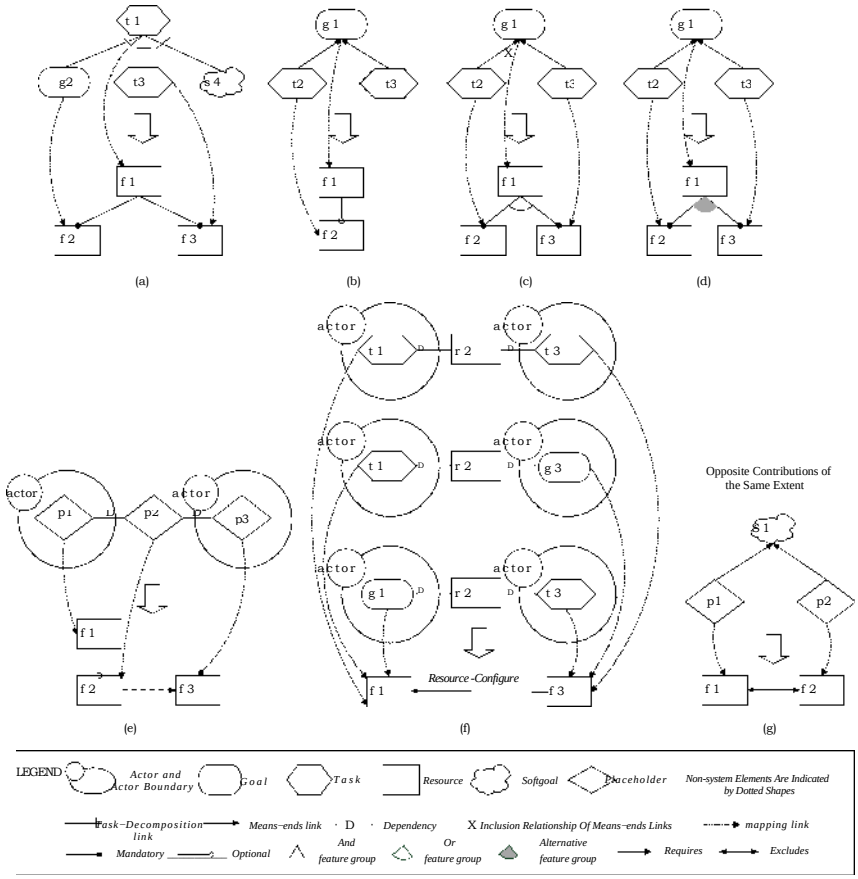

Fig. 1. Rules mapping between the i* framework and the feature model

3.2 Adjusting the initial domain feature model

After constructing initial domain feature model, we should further adjust the initial domain feature model according to our own experience and understanding in the domain. The initial domain feature model merely contains the feature trees which do not have a common root node. Therefore, we regard the whole system as the root node and merge each feature tree as its child respectively. In addition, we modify the features' names in order to make them more illustrative, and meantime remove the redundant features if necessary.

4 Case studies

Our proposed approach is successfully applied in the Hearing-aids Sales and Distribution Software Product Line (HSDSPL), whose i* model is described in Fig. 2.

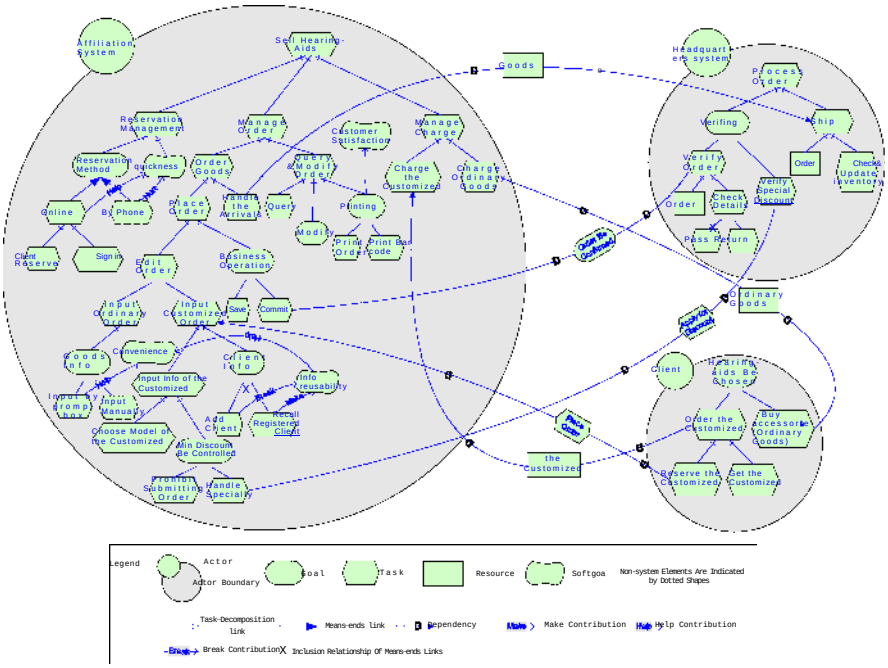

Fig. 2. The i* model of HSDSPL

There are three actors representing the division system, the headquarters system and the client respectively. Since Actor Client is a non-system element, it cannot be mapped in the feature model. Actor Division is responsible for selling hearing-aids and accessories supplied by the headquarters, whereas Actor Headquarters is responsible for the production of hearing-aids and accessories. During the actual operation, the divisions send orders to the headquarters. After the orders are examined and confirmed, the headquarters produces and delivers the customized goods or just delivers the off-the-shelf goods directly to divisions. The domain feature model corresponding to Fig. 2 based on our approach is shown in Fig. 3.

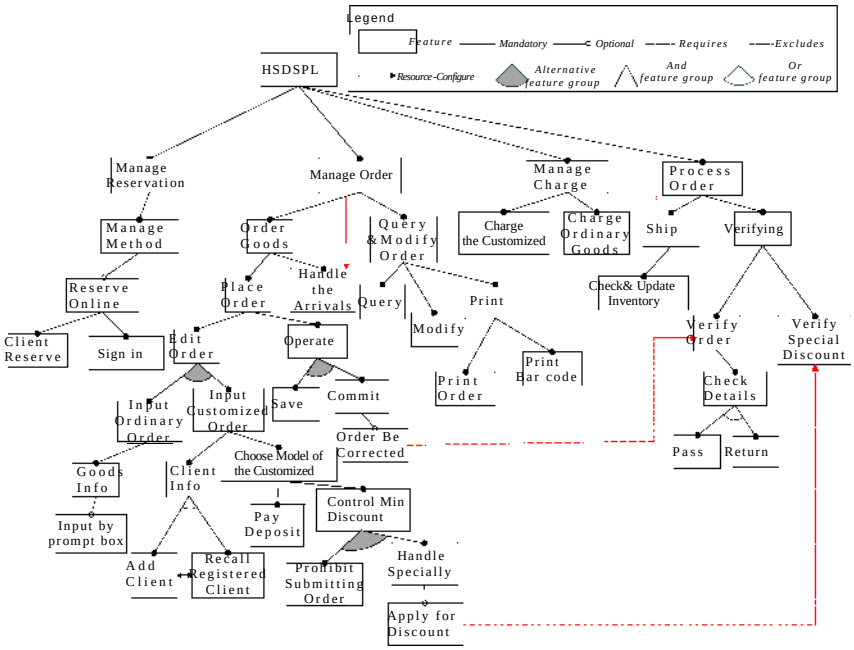

Fig. 3. The domain feature model of HSDSPL

5 Conclusions

In this paper, we adopt the i* framework to construct domain feature model. We propose the algorithm to map the dependencies among actors to feature models. Besides, we initiate to customize an application product using the i* model evaluation during the application engineering phase.

Sometimes, the softgoal dependencies among actors involve other softgoals, and are shown in different situations. To the best of our knowledge, however, it's difficult to propose rules mapping softgoal dependencies among actors. We intend to further study the approach and propose a more complete solution in the future. In addition, combining our approach with the scenario method for feature dependencies study to guide the architectural design is also on the list of our future work.

Acknowledgments. The work is supported by the Key Science and Technology Project of Zhejiang (No. 2012C11026-3, No. 2008C11099-1) and the open project foundation of Zhejiang Provincial Key Laboratory of Network Technology and Information Security. The authors would also like to thank anonymous reviewers who made valuable suggestions to improve the quality of the paper.

References

1. K.Pohl, G.Böckle, and Frank van der Linden: Software Product Line Engineering Foundations, Principles, and Techniques. Springer Verlag, Heidelberg (2005)
2. Kang, K.C., Lee, J., and Donohoe, P.: Feature-Oriented Product Line Engineering. IEEE Software 19, pp. 58-65 (2002)
3. W. Zhang, H. Mei, and H. Zhao: Feature-driven Requirement Dependency Analysis and High-level Software Design. In Requirements Engineering, vol. 11, no. 3, pp. 205-220 (2006)
4. Yu. Towards modeling and reasoning support for early-phase requirements engineering. In Proceedings of the IEEE International Symposium on Requirement Engineering, pp. 226–235. IEEE Computer Society (1997)
5. E. Yu and J. Mylopoulos: Understanding why in software process modeling, analysis, and design. In Proc. 16th Intl. Conf. on Soft. Eng., May 16-21, pp. 159–168 (1994).
6. Horkoff, J.: Using i* modeling for evaluation, Master Thesis, University of Toronto, Department of Computer Science (2007)
7. Horkoff, J., Yu, E.: Evaluating Goal Achievement in Enterprise Modeling: An Interactive Procedure and Experiences. In: The Practice of Enterprise Modeling. Springer, pp. 145160 (2009)
8. Yu, Y., Leite, J. C. S. D. P., La pouchnian, A. and Mylopoulos, J.: Configuring features with stakeholder goals. In: Proc. ACM Symp. on Applied Computing , pp. 645-649 (2008)

