Building Requirements Traceability to Support Specification Construction and Evolution

Weichen Cail, Shaoying Liu[2]

Graduate School of Computer and Information Sciences, Hosei University
Tokyo 184-8584, Japan
Department of Computer Science, Hosei University
Tokyo 184-8584, Japan

weichen.cai.jh@stu.hosei.ac.jp, sliu@hosei.ac.jp

Abstract. Requirements traceability has been widely recognized as an indispensable support to software development activity, especially during recent years, with the scale of the software system becoming unprecedentedly larger and the time period of developing a software system becoming longer. This situation brings up more evolution in requirements specifications and makes requirements traceability more attractive. However, due to the informality of the specifications, the traceability of requirements during evolutions is unlikely to be built systematically and precisely. In this paper, we take advantage of the SOFL formal specification language and its three-step modeling approach to establish an effective approach to building requirements traceability systematically.

Keywords: Requirements traceability, formal specification, documentation evolution

1 Introduction

Iterative approach for software development has been widely accepted and used in industrial practice for a long time [1]. Using an iterative approach always come with changes in the documents we create and have to maintain during the whole software development life-cycle. These changes in the documents are called documentation evolution. A great challenge in documentation evolution is how to sustain the consistency between different level documents.

Inconsistency caused by documentation evolution will happen between either homogeneous documents created during the same development phase or heterogeneous documents created during different development phases. Such inconsistency between documents will deteriorate the trust to these documents by the designers, analysts, developers and clients which will finally make most of the documents less useful [2][3]. Documentation traceability is mainly focusing on solving the inconsistency during the document evolution. By creating links between different documents, the changes in each document are traceable.

Of many existing formal methods, the SOFL formal engineering method has proved to be a practical and effective method for its three-step modeling approach [4]. Developing a software system using SOFL *(Structure Object-oriented Formal Language),* the starting point is to construct an informal specification that includes user requirements for functions, data resources, and constraints. To clarify the semantics of the descriptions in the specification and configure the relation between different parts, the informal specification is transformed into a semi-formal specification. This transformation is usually an evolutionary process due to the nature of human understanding. Finally, the semi-formal specification is transformed into a formal design specification in which all of the data and functions are defined using a mathematically-based notation.

To ensure that every requirement presented in the informal specification is correctly defined in the semi-formal specification and then further correctly realized in the formal design specification, requirements traceability must be built and utilized to rigorously check the consistency between different level specifications, but how to systematically build the traceability is still an open problem.

In this paper, we focus on this problem and propose an approach to build requirements traceability as a solution. The essential idea of our approach is that every item in the informal specification, which represent user's functional or nonfunctional requirement, must be linked to one or more items in the semi-formal specification, while every item in the semi-formal specification must be linked to one or more items in the formal design specification in the same principle.

The rest of the paper is organized as follows. In Section 2, we discuss the corresponding relationship between elements within each specification in detail. Related work is introduced in Section 3. Finally, we give conclusions and point out future research directions in Section 4.

2 Traceability between specifications

Creating traceability between the three level specifications means connecting the elements in different format within different level specifications which are all represented to the same user requirements. For example, a data resource in the informal specification may be connected to a type identifier declaration and a state variable declaration in the corresponding semi-formal specification. In this section, we discuss how to link (or connect) elements in one specification to those in another.

2.1 Traceability between informal and semi-formal specifications

During the transformation from an informal specification to a semi-formal specification, the user requirements will be refined and described more precisely. Because of this refining process, the elements used to express the user requirement in informal specification are no longer sufficient. These elements within informal specification will be converted into several kinds of elements in semi-formal specification. Even such conversion is quite flexible and mainly depends on the user's judgment, it still only can be done between the elements represent homogeneous requirements. The convertible elements corresponding to the elements in informal specification are shown in Table 1.

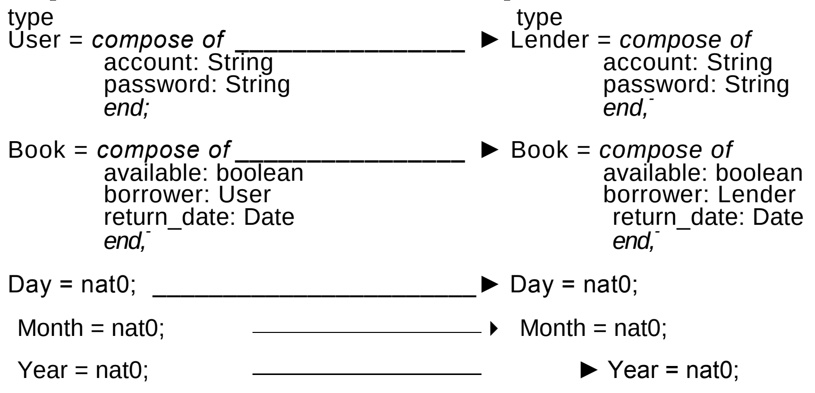
Table 1. The acceptable links are shown in Table table related

Informal Specification	Semi-formal specification
Function	Function
	Process
	Module
Data resource	Type identifier
	Constant identifier
	State variable
Constraint	Invariant

In Table 1., we put the elements in informal specification on the left side while its corresponding elements in semi-formal specification are allocated on the right side of Table 1.

2.2 Traceability between semi-formal and formal specifications

During the conversion from semi-formal to formal specification, there is no change on the kind of constitutional elements of both specifications which means that the corresponding relationship can only exist between the same kind of elements within semi-formal and formal specification, as it is theoretically unconvertible between different kinds of elements. The traceable links created between type identifier declaration part between semi-formal and formal specification is shown in Fig. 3.



```
type                                          type
User = compose of ____________________ ► Lender = compose of
          account: String                          account: String
          password: String                         password: String
          end;                                     end,

Book = compose of ____________________ ► Book = compose of
          available: boolean                       available: boolean
          borrower: User                           borrower: Lender
          return_date: Date                         return_date: Date
          end,                                      end,

Day = nat0;  __________________________► Day = nat0;

 Month = nat0;          ________________ ▸  Month = nat0;

 Year = nat0;           ________________        ► Year = nat0;
```

Fig. 1. Trace links between two type identifier declaration within semi-formal and formal specification respectively

3 Related work

A large amount of publications have affirmed that requirement traceability plays an essential role in software engineering [5][6]. However, once the problem come to how the trace information can be create and maintain, the lists become very short.

Works of Gotel and Finkelstein [5] have done an investigation on the traceability problem and discuss the reason for which it still exists. One reason they stated is that the lack of pre-requirements traceability. They argue that tracing requirement is not enough since the requirement may not be capture rationale. We agree with this point, however, we also believe that extra time consumption cause by manually trace information creation also limits the adoption of the current approach in the industrial practice.

Lange and Nakamura introduced an approach for tracing and visualization [7]. They draw out the run-time behaviors of software system to capture the trace information, but they focus more on how to visualize trace information than generate or validate trace information. This approach starts after development phase, while our approach focus more on the requirement analysis and system design phase within the software development life-cycle.

Easterbrook and Callahan [8] introduced an approach for verification and validation of specification using formal method. In their works, they introduced an AND/OR table as an intermediate to relate textual requirements and SCR model. However, they didn't give the concrete process of the trace information generating. Despite some deficiencies of the approach we mentioned above, all techniques discussed are useful since they cover the software development approach and requirement modeling techniques outside formal method domain.

4 Conclusion

In this paper we proposed an approach supporting the generation of trace information by using SOFL method. We briefly introduced the three heterogeneous requirement specifications derived from SOFL method. We then go through the corresponding relationship among informal, semi-formal and formal specification.

The key contribution of our approach is that we define the acceptable linking relationship between the heterogeneous elements within different SOFL specifications.

Further work will concentrate on providing a semi-automatic or automatic approach to reduce the effort and time cost on creating trace link in a manually manner. Also a tool will be developed to support our approach to benefit the SOFL method user.

Acknowledgment

This research is supported in part by SCAT research foundation and Hosei University. It is also partly supported by China 973 program under Grant No. 2010CB328102 and National Natural Science Foundation of China (NSFC) under Grant Nos. 61133001, 60910004.

References

1. Sooriamurthi, R., Introduction to Programming and Software Development: an Interactive, Incremental and Iterative Approach, Proceedings of the 36th Annual Meeting of the Decision Sicences Institute DSI-2005, San Francisco, California, pp 1851-1856
2. Haumer, P., Pohl, K., Weidenhaupt, K., and Jarke, M., Improving Reviews by Extending Traceability, Proceedings of the 32nd Annual Hawaii International Conference on System Sciences (HICSS), 1999.
3. Clarke, S., Harrison, W., Ossher, H., and Traa, P. Subject-Oriented Design: Towards Improved Alignment of Requirements, Design, and Code, Proceedings of 1999 ACM SIGPLAN Conferencer on Object-Oriented Programming, Systems, Languages, and Applications, Dallas, TX, October 1998, pp.325-339
4. S. Liu, Formal Engineering for Industrial Software Development Using the SOFL Method, Springer-Verlag, 2004.
5. Gotel, 0. C. Z. and Finkelstein, A. C. W., An Analysis of the Requirements Traceability Problem, Proceedings of the First International Conference on Requirements Engineering, 1994, pp. 94-101.
6. Watkins R. and Neal M., Why and How of Requirements Tracing. IEEE Sopare 11(4), 1994, 104-106.
7. Lange D. B. and Nakamura Y. Object-Oriented Program Tracing and Visualization. IEEE Computer
8. Easterbrook S.; Callahan J., Formal Method for Verification and Validation of partial specification: A Case Study, Journal of Systems and Software, Volume 40, Number 3, March 1998, pp. 199-210(12)