In Wolfgang Hesse and Andreas Oberweis, editors, SIGSAND-EUROPE 2008 Proceedings of the Third AIS SIGSAND European Symposium on Analysis, Design, Use and Societal Impact of Information Systems, volume 129, pages 75–77, Berlin, 2008. Springer LNI.

OOLH: A formal framework for specifying system requirements

Yuen Man Hon, Jan-Tecker Gayen
Institute of Railway System Engineering and Traffic Safety
Technical University of Braunschweig, Germany
y.hon@tu-bs.de, j.gayen@tu-bs.de

Hans-Dieter Ehrich
Institute of Information Systems
Technical University of Braunschweig, Germany
HD.Ehrich@tu-bs.de

Abstract: Most of the system requirements are written in natural language. It is not easy for the system development team to understand this document unambiguously without domain specific knowledge. It is difficult to check the correctness of these requirements. A formal framework called Object Oriented Lastenheft (German for requirements specification) (OOLH) is proposed as a solution to handle these problems [1]. This framework provides well-defined mathematical concepts to formulate system requirements. These well-formalized system requirements can be analyzed and understood easier and their consistency can be checked based on the mathematical concepts. A tool, called OOLH tool, is implemented to support analyzing, verifying and checking consistency of formulas in OOLH. Logical formulas can be transformed into decision tables and truth tables. The expected behavior or a design can be specified in decision tables in this tool, such that the correctness of requirements can be verified.

Keywords: formal system requirements, consistency, correctness, analysis, verify

1 Introduction

There are two possible disadvantages in writing system requirements in natural language. They are ambiguity of specifications and checking the correctness of the specifications [SS97]. Applying formal methods to specify the system requirements is one of the solutions to reduce these disadvantagess [Win90]. A formal framework for specifying system requirements called OOLH is proposed. In OOLH, the system requirements written in natural languages are specified with mathematical concepts, such that the ambiguity of the requirements can be eliminated, their understandability can be improved and their consistency can be checked. As a result, the efficiency of developing systems can be increased.

[1] The concepts of the framework are developed within a research project of specifying the system requirements of German regional computerized railway interlocking systems called **L**astern**h**eft **ESTW-R**. The framework is named after this research project.

Furthermore, a tool, called OOLH tool, is developed based on these mathematical concepts to support analyzing these well-formed specifications. In this paper, the concepts of OOLH in supporting specification, analysis and verification are introduced.

2 OOLH: Specification, Analysis and Verification

OOLH is proposed to handle the problems of specifying system requirements in natural language. Most of the system requirements consist of mainly two types of requirements. They are static and dynamic requirements. Dynamic requirements describe the phases of the system's life cycle and relations among these phases. In each of the phase, static conditions need to be fulfilled. Furthermore, a system contains objects. Objects have attributes and actions. In OOLH, propositional logic is used to define the static conditions, while temporal logic and state machines are used to specify the dynamic and sequence oriented conditions [HR04]. During the translation of the system requirements into logics, the objects of the system and their attributes are defined. After the requirements are defined based on the attributes of the objects and logics, they are then understood easier and unambiguously.

It is not easy for non-software professionals to understand the semantics of logical requirements, as a result, formulas must be transformed into a form that can help them to analyze the meaning of the formula. This problem has been addressed in the specification language, Requirements State Machine Language (RSML) [HL96]. However, their tabular forms are not easy to read because they are the representation of the formula in DNF. It has been shown that decision tables are a suitable form to specify requirements and express knowledge [GN95, Van05]. One of the advantages of using decision tables as a specification method is that the requirements can be expressed in a compact form in a decision table by combining rules. Therefore, in OOLH, propositional formulas can be transformed to decision tables to increase the understandability of formal requirements. **O**rdered **B**inary **D**ecision **Diagrams** (OBDDs) are used to represent a propositional formula [Bry86]. A compact decision table can be automatically generated from the OBDD and the semantics of the formula is preserved.

A requirement can be transformed into a decision table, so that domain experts can check the correctness of the specification in a different view. Furthermore, users can specify their expected behaviors in formulas or decision tables. These expected behaviors can be checked against the specified requirements in OOLH. If the correctness of the requirements has been verified, these well-formed formulas can be used to verify the artifacts that are produced during the system development process. In the OOLH tool, a design can be specified as a decision table or formula, it can be checked against the well-formed requirements. The OOLH tool supports checking the consistency and completeness of decision tables. This structure analysis is done based on the manipulation of boolean functions and OBDD is used as the implementation technique in the tool [Dre02]. Similar ideas have been applied in rule base verification [MV04].The theoretical background of using OBDDs for transformations and verification in the OOLH tool can be found in [Hon08].

2.1 Conclusion

In this paper, a formal framework called OOLH is introduced. It can be used to specify requirements that are written in natural language. By using OOLH, the understandability of the requirements is increased. It provides a chance for the members of the system development team to analyze the logical requirements by transforming them directly to decision tables. Furthermore, the consistency and correct behavior of the requirements can also be checked in OOLH. The main contribution of this formal framework is that a chance for non-software professionals to specify their requirements formally and check the correctness of the requirements is provided. A tool is implemented to achieve these goals. Railway Domain experts find that it is easier to understand and analyze the requirements and railway concepts via the transformation of the logical requirements to decision tables in the OOLH tool. A case study has also been done to find inconsistencies and incompleteness by using the OOLH tool. However, the current version of OOLH does not support specifying requirements with arithmetics expression. This needs further investigation. Checking the possibility in specifying dynamic conditions by temporal logic and state machines and searching a proper format to illustrate the semantics for the users to analyze these conditions are also included in the next step of this research work.

References

[Bry86] R. E. Bryant. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Transactions on Computers*, 35(8):677–691, 1986.

[Dre02] R. Drechsler. JADE: Implementation and Visualization of a BDD Package in JAVA, 2002. citeseer.ist.psu.edu/503345.html.

[GN95] Pu H.-C.and Rom W.O. Gorla N. Evaluation of process tools in systems analysis. *Information and Software Technology*, 37(2):119–126(8), 1995.

[HL96] M. P. E. Heimdahl and N. G. Leveson. Completeness and Consistency in Hierarchical State-Based Requirements. *Software Engineering*, 22(6):363–377, 1996.

[Hon08] Y.M. Hon. Notes on Decision Tables and OBDD. Internal Report, 2008. http://www.tu-braunschweig.de/ifev/forschung/projekte/specestw.

[HR04] M Huth and M. Ryan. *Logic in Computer Science*. Prentice Hall, 2 edition, 2004.

[MV04] C. Mues and J. Vanthienen. Efficient Rule Base Verification Using Binary Decision Diagrams. In *Database and Expert Systems Applications*, volume 3180, pages 445–454. Springer, Berlin/Heidelberg, 2004.

[SS97] I. Sommerville and P. Sawyer. *Requirements Engineering: A Good Practice Guide*. John Wiley and Sons, 1997.

[Van05] J. Vanthienen. Consistency by Construction: Decision Table Experiences in Business Rules and Business Processes. In *European Business Rules Conference*, pages 67 – 74, June 2005.

[Win90] J.M. Wing. A Specifier's Introduction to Formal Methods. *IEEE Computer*, 23:8–24, 1990.