A Blame-Based Approach to Generating Proposals for Handling Inconsistency in Software Requirements

Kedian Mu, Peking University, China
Weiru Liu, Queen’s University Belfast, UK
Zhi Jin, Peking University, China

ABSTRACT

Inconsistency has been considered one of the main classes of defects in software requirements specification. Various logic-based techniques have been proposed to manage inconsistencies in requirements engineering. However, identifying an appropriate proposal for resolving inconsistencies in software requirements is still a challenging problem. This paper proposes a logic-based approach to generating appropriate proposals for handling inconsistency in software requirements. Informally speaking, given an inconsistent requirements specification, the authors identify which requirements should be given priority to be changed for resolving the inconsistency in that specification, by balancing the blame of each requirement for the inconsistency against its value for that requirements specification. The authors follow the viewpoint that minimal inconsistent subsets of a set of formulas are the purest forms of inconsistencies in that set. According to this viewpoint, a potential proposal for resolving inconsistencies can be described by a possible combination of some requirements to be changed that can eliminate minimal inconsistent subsets. Then a method is proposed of evaluating the degree of disputability of each requirement involved in the inconsistency in a requirements specification. Finally, an algorithm is provided of generating appropriate proposals for resolving the inconsistency in a given requirements specification based on the degree of disputability of requirements.

Keywords: Degree of Disputability, Inconsistency Handling, Prioritized Knowledge Base, Software Requirements, The Blame for the Inconsistency

1. INTRODUCTION

It has been increasingly recognized that inconsistency is inevitable during the requirements process (Easterbrook & Chechik, 2001a; Nuseibeh et al., 2001). Both general principles of managing inconsistency and special case-based approaches to handling inconsistency have recently been considered. In particular, it has been pointed out in Gervasi and Zowghi (2005) that the use of logic in managing inconsistency

DOI: 10.4018/jkss.2012010101
in requirements has been found to be effective in a number of studies. Various logic-based techniques have been proposed to manage inconsistencies in requirements engineering (Hunter & Nuseibeh, 1998; Gervasi & Zowghi, 2005; Martinez et al., 2008; Zowghi & Gervasi, 2003; Mu et al., 2005a, 2008, 2009). Most of these logic-based approaches focus on how to manage inconsistency by applying logical techniques such as paraconsistent reasoning and non-monotonic reasoning to requirements engineering. For example, Hunter and Nuseibeh (1998) developed the labeled quasi-classic logic to represent and reason about requirements specifications in the presence of inconsistency. Gervasi and Zowghi (2005) proposed methods for reasoning about inconsistencies in natural language requirements by combining natural language parsing techniques and non-monotonic reasoning. Easterbrook and Chechik (2001b) presented a framework termed χbel for merging inconsistent viewpoints using multi-valued logics. This framework was intended to highlight the source of inconsistency and to tolerate inconsistencies between viewpoints during model checking.

In contrast, there are relatively few logic-based techniques for generating appropriate proposals for inconsistency resolving actions in requirements engineering (Finkelstein et al., 1994; Gabbay & Hunter, 1993; Mu & Jin, 2007; Mu et al., 2008, 2009). Previously, we have argued that the relative priority of each requirement should play an important role in identifying appropriate proposals for resolving inconsistencies in requirement specifications (Mu & Jin, 2007; Mu et al., 2008, 2009), moreover, negotiation and combinatorial vote may be considered as two appropriate mechanisms of group decision making for identifying acceptable common proposals for handling inconsistent requirements specification (Mu et al., 2008, 2009). However, identifying appropriate actions for resolving inconsistency in requirements specification is still a challenging problem (Hunter & Nuseibeh, 1998). Generally, the choice of inconsistency handling actions is a rather context-sensitive issue (Finkelstein et al., 1994; Gabbay & Hunter, 1993). So, as pointed out in Mu et al. (2008), a feasible proposal for inconsistency resolving should focus on pointing out which requirements to be changed rather than how to change these requirements.

Roughly speaking, all the requirements involved in inconsistencies can be considered disputable. Each of such requirements is a candidate for requirements to be changed during the process of inconsistencies resolving. However, in many cases in requirements engineering, not all the requirements involved in inconsistencies need to be changed to resolve inconsistencies. Intuitively, the choice of requirements to be changed should depend on the evaluation of the blame of each requirement for inconsistencies in requirements specifications as well as the evaluation of the value of each requirement. To address this, in this paper, we present an approach to generating appropriate proposals for resolving inconsistencies in requirements specifications. This approach focuses on identifying requirements to be changed to resolve inconsistencies by balancing the blame of each requirement for inconsistencies against its value to the system-to-be. Informally, we formulate requirements specifications as prioritized knowledge bases in classical logic. Then we adopt the approach to measuring the blame of each formula for inconsistent prioritized knowledge bases presented in Mu et al. (2011) to evaluate the blame of each requirement for inconsistencies in an individual requirements set. Following this, we measure how disputable an individual requirement involved in inconsistency is by balancing the blame of that requirement against its priority. Finally, we propose an algorithm of choosing requirements to be changed based on this measurement.

The rest of this paper is organized as follows. Section 2 gives a brief introduction to the logical representation of requirements. We propose an approach to measuring how disputable a requirement involved in inconsistency is by balancing the blame of each requirement against its priority in Section 3. Section 4 proposes an algorithm of choosing requirements to be changed based on the degree of disputability.
Related Content

A Tool for Adaptive E-Assessment of Project Management Competences
www.igi-global.com/chapter/tool-adaptive-assessment-project-management/56077?camid=4v1a

A Lyapunov Theory-Based Neural Network Approach for Face Recognition
Li-Minn Ang, King Hann Lim, Kah Phooi Seng and Siew Wen Chin (2010). *Intelligent Systems for Automated Learning and Adaptation: Emerging Trends and Applications* (pp. 23-48).
www.igi-global.com/chapter/lyapunov-theory-based-neural-network/38449?camid=4v1a

Bigraphical Reactive Systems Based Approaches for Modeling Context-Aware Systems
www.igi-global.com/article/bigraphical-reactive-systems-based-approaches-for-modeling-context-aware-systems/120651?camid=4v1a
Parametric Generator for Architectural and Urban 3D Objects
www.igi-global.com/chapter/parametric-generator-architectural-urban-objects/69466?camid=4v1a