Journal of Visual Languages and Computing 26 (2015) 42–65

Contents lists available at ScienceDirect

Journal of Visual Languages and Computing

journal homepage: www.elsevier.com/locate/jvlc

Consistency of UML class, object and statechart diagrams using ontology reasoners ☆

Ali Hanzala Khan *, Ivan Porres

Department of Information Technologies, Åbo Akademi University Joukahaisenkatu 3-5, FI-20520 Turku, Finland

ARTICLE INFO

Article history:
Received 15 March 2014
Received in revised form
9 May 2014
Accepted 19 November 2014
Available online 26 November 2014

Keywords:
Consistency
Ontology
UML
Reasoning

ABSTRACT

We propose an automatic approach to analyze the consistency and satisfiability of Unified Modeling Language UML models containing multiple class, object and statechart diagrams using logic reasoners for the Web Ontology Language OWL 2. We describe how to translate UML models in OWL 2 and we present a tool chain implementing this translation that can be used with any standard compliant UML modeling tool. The proposed approach is limited in scope, but is fully automatic and does not require any expertise about OWL 2 and its reasoners from the designer.

1. Introduction

Model Driven Engineering (MDE) [1] advocates the use of models to represent the most relevant design decisions of a software development project. A MDE software development project involves the creation of many models. Each model is used for describing, visualizing and observing different viewpoints of a system at different levels of abstractions [2].

A software model usually comprises a number of diagrams. The diagrams in a software model are described using a particular modeling language. A well-known general modeling language used by practitioners during software development process is the Unified Modeling Language (UML) [2,3]. The definition of a modeling language is given in terms of a metamodel by using a metamodeling language, such as Meta Object Facility (MOF) [4] or Kernel Meta Meta Model (KM3) [5]. This paper focuses on the analysis of models specified using UML superstructure specification [2] and MOF.

UML models can be represented in the form of a theory in mathematical logics [6], such as, description logics [7] or predicate logics [8]. A *consistent* logical theory is the one which does not contain a contradiction or an unsatisfiable concept [9,10]. Similarly, we consider a model to be consistent if it does not contain an unsatisfiable concept.

The presence of concepts in a model that are not satisfiable reveals design errors. For example if a UML class diagram depicts unsatisfiable classes, then it is not possible to instantiate objects conforming to theses classes. Furthermore, in case of an inconsistent behavioral diagram (such as inconsistent statechart diagrams), an object cannot enter into an unsatisfiable state.

The unsatisfiable concepts in models should be identified as early in the development process as possible. In this paper we propose an approach that automatically checks the consistency and satisfiability of UML models. If a model is found to be inconsistent, then the proposed approach indicates the unsatisfiable concepts that make the whole model inconsistent.

We call the task of finding out the inconsistencies in software models a *model validation*. The validation of modeling artifacts has been discussed in many research papers. However, we consider that there is still need for more

☆ This paper has been recommended for acceptance by Shi Kho Chang.

* Corresponding author. Tel.: +358 2 215 3463.
E-mail addresses: ali.khan@abo.fi (A.H. Khan), ivan.porres@abo.fi (I. Porres).

http://dx.doi.org/10.1016/j.jvlc.2014.11.006
1045-926X/

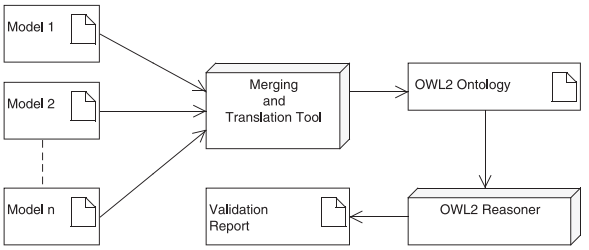

Fig. 1. Workflow of the proposed consistency checking approach.

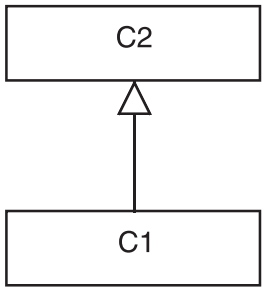

Fig. 2. A UML model depicting a class C1 being a subclass of a class C2.

research in this area due to the vast number of different validation problems that exist for complex models.

Among the validation problems in behavioral diagrams that has already been discussed by other researchers there are for instance, analysis of the control looping to find deadlocks [11], analysis of method invocations against the class description for finding deadlocks [12]. Also, checking the consistency of statechart diagrams and class diagrams by using the π-calculus [13]. The research on validation problems of structural diagrams is also very vast. A number of problems that have been discussed in the recent past by other researchers include the consistency of UML class diagrams with hierarchy constraints [14], the reasoning of UML class diagrams [15], the full satisfiability of UML class diagrams [16], and the inconsistency management in model driven engineering [17]. Although, a lot of research work has already been done in the area of the validation of structural and behavioral diagrams, we still believe there is a room for new approaches in this area.

The validation problem that we tackle in this paper can be stated as follows: Is a model containing multiple UML class diagrams, UML object diagrams and UML statecharts containing class and state invariants consistent and satisfiable? Model consistency and satisfiability is established by translating the models into a logical theory, and then using automatic logical reasoners to infer the logical consequences of the translated models. More concretely, we propose to represent UML models using a description logic by means of the OWL 2 DL language [18,19], and to analyze the translated models using automated OWL 2 reasoners [20,21]. The approach we present in this paper is fully automated and it is implemented in the form of a tool that can be used with any standard compliant UML modeling tool.

In order to implement a fully automatic tool, we have decided to use description logic as the underlying formalism for our approach and OWL 2 DL [19] as the language to represent the UML models internally. This decision is supported by the fact that there are reasoners for description logic with the efficient decision procedures that are automatic [20,21]. Alternatively, there is a number of theorem proving tools available that are based on a high order logic, such as HOL [22]. These tools are very powerful but they require interaction with an expert human user. Also, there are model finders, such as Alloy [23] or Microsoft formula [24], which are automatic, but require that we artificially limit the search space.

The workflow of our approach is shown in Fig. 1. A number of UML models are taken as an input. All the inputs are translated to a decidable fragment of OWL 2, i.e., OWL 2 DL [19]. In the next step, the OWL 2 translations of UML diagrams are passed to a reasoner in the form of an ontology. The reasoner processes the ontology and produces a validation report. The validation report reveals the inconsistencies in the ontology representing the UML models. The detailed discussion about the contents of the validation report is discussed later in different sections of this paper.

In this paper we address the issues that have been inadequately or not addressed in the previous research. The novelty of our work is that we offer the validation of many modeling concepts under one approach. The modeling concepts that can be validated using our approach include the following: classes, objects, associations, links, labeled links, domain and range, multiplicity, composition (herein unshearedness and acyclicity), unique and non-unique associations, ordering, class generalization, and association generalization. Furthermore, the proposed approach also allows us to analyze the conformance of object diagrams against class diagrams, consistency of class diagrams and statchart diagrams, consistency of state invariants written using a subset of object constraint language, and the consistency of multiple models when merged together.

This paper also contains several example applications of the proposed approach. These include (1) validation of multiple models of the same metamodel when merged, (2) validation of class and object diagrams with OCL invariants and (3) validation of class and statechart diagrams with OCL state invariants. The detection of errors in the above-mentioned models, and the results of the performance tests of the proposed approach that is shown in this paper is the evidence that the proposed approach is viable and practical, and can be applied in the industry.

In the next section we give an overview of this research.

2. Background

2.1. Ontology foundations

An ontology is a specification of a conceptualization [25]. In this paper our understanding of the term "*specification of a conceptualization*" is the specification of concepts and relationships that can represent an abstraction of a program. The abstraction of a program is typically expressed in the form of models by using modeling languages. For example the fact that a class C1 is a subclass of a class C2 is drawn by using UML, as given in Fig. 2.