Assessing Modal Aspects of OntoUML Conceptual Models in Alloy

Alessander Botti Benevides, Giancarlo Guizzardi, Bernardo F. B. Braga and João Paulo A. Almeida

Ontology and Conceptual Modeling Research Group (NEMO),
Computer Science Department, Federal University of Espírito Santo (UFES),
Av. Fernando Ferrari, n 514, Goiabeiras, Vitória, ES, Brazil
`{abbenevides,gguizzardi,bfbbraga}@inf.ufes.br,jpalmeida@ieee.org`

Abstract. Assessing the quality of conceptual models is key to ensure that conceptual models can be used effectively as a basis for understanding, agreement and construction of information systems. This paper proposes an approach to assess conceptual models defined in OntoUML by transforming these models into specifications in the logic-based language Alloy. These Alloy specifications include the modal axioms of the theory underlying OntoUML, allowing us to validate the modal meta-properties of the OntoUML types and part-whole relations.

1 Introduction

John Mylopoulos [1] defines conceptual modeling as "the activity of formally describing some aspects of the physical and social world around us for purposes of understanding and communication". In this view, a conceptual model is a means to represent what modelers (or stakeholders represented by modelers) perceive in some portion of the physical and social world, *i.e.*, a means to express their conceptualization [2] of a certain universe of discourse.

If conceptual models are to be used effectively as a basis for understanding, agreement, and, perhaps, construction of an information system, conceptual models should express as accurately as possible a modeler's intended conceptualization. More specifically, the model should ideally describe all states of affairs that are deemed *admissible* and rule out those deemed *inadmissible* according to the conceptualization [2].

As argued for in [2], the quality of a conceptual modeling language can be assessed by considering the extent to which the language supports the definition of models that capture this intended conceptualization. This concern has justified the revision of a portion of UML into a conceptual modeling language named OntoUML. This revision enables modelers to make finer-grained distinctions between different types of classes and different types of part-whole relations according to the UFO foundational ontology [2].

More specifically, this revision introduces modal meta-properties for object classifiers that enable one to distinguish between rigid, semi-rigid and anti-rigid classifiers (and therefore distinguish properties that apply necessarily to objects from those that apply contingently) as well as meta-properties for part-whole relations to distinguish between mandatory, essential, inseparable and immutable parts, and immutable wholes [2].

Although the language revision impacts on a modeler's ability to express the intended conceptualization accurately, one would certainly be naive to assume that modelers make no mistakes while constructing the models and that they fully understand the theory that supports the language. These cases could lead to ill-defined conceptual models, which may be: (i) syntactically incorrect; (ii) syntactically correct, but unsatisfiable; (iii) syntactically correct, satisfiable, but invalid according to the intended conceptualization.

Previous efforts in addressing the assessment of OntoUML models have focussed on syntactic correctness and led to the specification of OntoUML's syntactical constraints as OCL expressions on the language's metamodel [3] and the building of a graphical editor [3] that is capable of automatic syntax verification. In this paper, we go beyond syntax verification and aim at addressing the satisfiability and validity of OntoUML models. More specifically, we discuss an approach based on formal specifications in the logic-based language Alloy [4] to generate instances of an OntoUML model with the purpose of showing that the model is satisfiable and improving the modeler's confidence in the validity of the model.

In our approach, the Alloy specification is fed into the Alloy Analyzer to generate a branching-time Kripke world structure [5] that reveals the possible dynamics of object creation, classification, association and destruction. Each world in this structure is an instance of the OntoUML model and represents a snapshot of the objects and relations that exist in that world. This world structure is necessary since the meta-properties characterizing most of the ontological distinctions in UFO are modal in nature. (For example, the definition of a "rigid" classifier states that it applies necessarily to its instances in all worlds in which they exist.) We have specified UFO's modal axioms in Alloy to guarantee that the generated world structure satisfies these axioms by construction. Therefore, the sequence of possible snapshots in this world structure will support claims of model satisfiability and improve our confidence on claims of validity.

This paper is further structured as follows. Section 2 presents the modal aspects of UFO's object types and part-whole relations and presents our running example; section 3 briefly describes the suitable Kripke world structures and their representation in Alloy; section 4 exemplifies a world structure generated by the Alloy Analyzer; section 5 discusses related work and, finally, section 6 presents our final considerations.

2 OntoUML Concepts

Due to space limitations, we concentrate here on a fragment of the Unified Foundation Ontology (UFO) [2], with a specific focus on those distinctions that are spawned by variations in meta-properties of a modal nature. UFO's main categories are depicted in Fig. 1 below and are briefly discussed in the remainder of this section by using a running example depicted in Fig. 2. Since OntoUML is a modeling language whose metamodel is designed to be isomorphic to the UFO ontology, the leaf ontological distinctions in Fig. 1 appear as modeling primitives in the language (see stereotyped classes and relationships in Fig. 2).

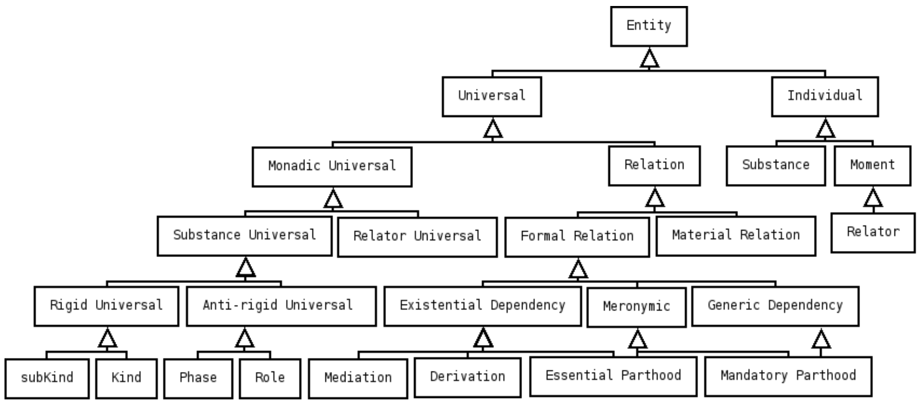

Fig. 1. Excerpt of UFO taxonomy [2].

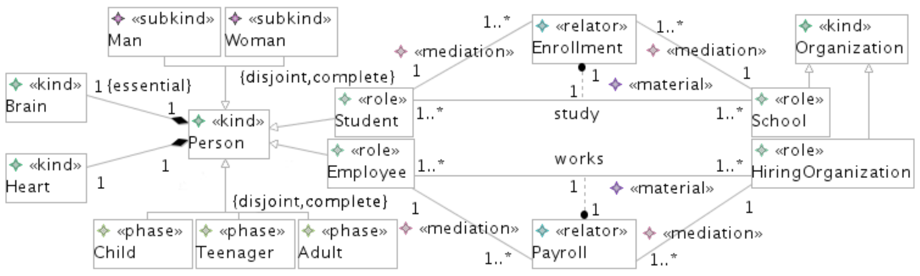

Fig. 2. Running example.

2.1 Substances and Moments

UFO is based on a fundamental distinction between Individuals and Universals (roughly instances and types, respectively) and, within the category of individuals, it differentiates between Substances and Moments. The distinction between Substances and Moments is based on the formal notion of existential dependence, a modal notion that can be briefly defined as follows: **Definition 1 (existential**

dependence): an individual x is *existentially dependent* on another individual y iff, as a matter of necessity, y must exist whenever x exists. In other words, in every world w, if x exists in w then y must also exist in w. ■

Substances are existentially independent individuals, *i.e.*, there is no Entity x disjoint from y that must exist whenever a Substance y exists. Examples of Substances include ordinary mesoscopic objects such as a Person or a Car. Conversely, a Moment is an individual that can only exist in other individuals, *i.e.*, that is existentially dependent on other individuals. Here, we concentrate on relational moments or relators (*e.g.*, a covalent bond, an enrollment or a marriage).

So, a Substantial Universal is a universal whose instances are Substances (*e.g.*, the universal Person or the universal Apple). While, a Relator Universal is a universal whose instances are individual relational moments (*e.g.*, the particular enrollment connecting *Alex* and a certain School is an instance of the universal Enrollment).

2.2 Substance Universals

We need to define some additional modal notions (rigidity and anti-rigidity) to be able to make further distinctions within Substance Universal. **Definition 2 (Rigidity)**: A universal U is rigid if for every instance x of U, x is necessarily (in the modal sense) an instance of U. In other words, if x instantiates U in a given world w, then x must instantiate U in every world w` accessible from $w$. ■ **Definition 3 (Anti-rigidity)**: A universal $U$ is anti-rigid if for every instance $x$ of $U$, $x$ is possibly (in the modal sense) not an instance of $U$. In other words, if $x$ instantiates $U$ in a given world $w$, then there must be a possible world $w$`, accessible from w, in which x does not instantiate U. ■

Substantial Universals that are rigid are named Kinds and subKinds. These universals define a stable backbone, a taxonomy of rigid universals instantiated by a given individual (the Kind being the unique top-most rigid universal instantiated by an individual).

Within the category of anti-rigid substantial universals we have a further distinction between Phases and Roles. Both Phases and Roles are specializations of rigid universals (Kinds/subKinds). However, they are differentiated w.r.t. their specialization conditions. For the case of Phases, the specialization condition is always an intrinsic one. For instance, in Fig. 2, a child is a Person whose age is within a certain range. For Roles, in contrast, their specialization condition is a relational one: a Student is a Person who is enrolled in (has a study relation to) a School, *etc.* Formally speaking, this distinction is based on a meta-property named Relational Dependence: **Definition 4 (Relational Dependence)**: A type T is relationally dependent on another type P via relation R iff in every world w, for every instance x of T there is an instance y of P in that world such that x and y are related via R in w. ■ Finally, as discussed in [2], Phases (in contrast to Roles) are always defined in a partition set. For instance, in Fig. 2, the universals Child, Teenager and Adult define a phase partition for the Kind Person. As consequence, we have that in an each world w, every Person is either a Child, a Teenager or an Adult in w and never more than one of these. Additionally, if x is a Child (Teenager, Adult) in w, there is always a possible world w', accessible from w, in which x will not be a Child, in which case he will be either a Teenager or an Adult.

In summary, in the example of Fig. 2, these model distinctions are exemplified by contrasting the (Kind) universal Person, the (Role) universal Student and the (Phase) universal Teenager.

2.3 Relator Universals and Relations

In order to represent the relation between Student and Person, one should model Student as a Role played by Person in a certain context, where he is enrolled in a School. Analogously, one should model School as a Role played by an Organization when providing educational services to a Student. This context is materialized by the Material Relation *study* (represented as the «material» stereotype in OntoUML), which is in turn, derived from the existence of the Relator Universal Enrollment («relator»). In other words, we can say that a particular student x studies at a particular school y iff there is an Enrollment z that mediates x and y. This situation is illustrated in Fig. 2. The formal relations of mediation in this model represents the existential dependence of the relator on its bearers [2].

Once more, we concentrate here on two different types of part-whole relations that are distinguished based on modal meta-properties, in particular, the previously defined notions of Existential and Relational Dependence. As one can observe contrasting the Definitions 1 and 4, the former is a relation between two individuals, whilst the latter is a relation between types. An Essential Parthood relation is a parthood relation that implies existential dependence. Contrariwise, a Mandatory Parthood relation is one that implies (generic) relational dependence (where the relation R defined in Definition 4 is a relation of formal parthood). These two types of relations are exemplified in Fig. 2 by the relations Brain-Person and Heart-Person, respectively. An Essential Parthood relation between the universals Brain and Person implies that: for every x instance of Person there is an individual y instance of Brain such that x cannot exist without having that specific Brain as a part (*i.e.*, y cannot change from world to world). The mandatory parthood between the universals Heart and Person instead implies that: for every x instance of Person, x cannot exist without a (generic) instance of Heart as a part.

3 Representing Modality in Alloy

Our approach is based on the transformation of OntoUML into Alloy. The product of this transformation is an Alloy specification that can be fed into the Alloy Analyzer to generate a Kripke structure and its associated worlds that together respect UFO's (modal) axioms. This allow us to show that an OntoUML model is semantically consistent (satisfiable). Furthermore, we believe that the analysis of a well-chosen set of these structures (*e.g.*, structures that exhibit important behavior of model's instances) can improve the modeler's confidence in the validity of the model.

We represent modality explicitly in the generated Alloy specification. This means that this specification reifies the notion of Kripke world structure and explicitly manipulates the accessibility relations between worlds. This is necessary to specify UFO's modal axioms, given that no notion of modality is built-in in Alloy.

Our intention is to represent a possible worlds structure in which the accessibility relations between worlds represents the common sense temporal structure. In our ordinary language, we are able to talk about the present, the past, the possible future, and the facts that could have happened, but accidentally did not (*i.e.*, the counterfactuals). So, in this work, we consider a Kripke structure that is able to handle all these notions.

The transformation we have implemented maps OntoUML classes to Alloy signatures and OntoUML binary relationships to Alloy ternary relations declared inside signatures. For example, an OntoUML relation R that relates A to B is mapped to an Alloy relation $R =< A, B, World >$, where the third field denotes the worlds in which the relationship exists. The characteristics of the OntoUML classes (*e.g.*, rigidity, anti-rigidity) and the ones of the relationships (*e.g.*, cardinality constraints, shareability, existential dependency and disjointness) are mapped to constraints in Alloy.

Listing 1.1 shows an excerpt of the Alloy specification that was automatically generated from the model depicted in Fig. 2. This specification illustrates the representation of the OntoUML class Person in Alloy.

Listing 1.1. Alloy model excerpt

```
1 sig Person_Set in Concept { Person: some World }
2 {
3   Person in existsIn
4   all w1: World | w1 in Person => (all w2:
        w1.access | (w2 in existsIn) => (w2 in
        Person)) -- Rigidity
5   some w: World | w in this.Child -- Phase
6   some w: World | w in this.Teenager -- Phase
7   some w: World | w in this.Adult -- Phase
8   ⋮
9 }
```

This excerpt shows the specification of the Rigidity of the Kind Person (line 4) and the possibility of a Person to be an instance of Child, Teenager or Adult (lines 5, 6 and 7, respectively).

4 Generating Instances

Fig. 3 shows the Kripke structure and associated worlds that are generated by the Alloy Analyzer for the model shown in Fig. 2. This structure demonstrates the satisfiability of the model and exemplifies some of its dynamic aspects, assisting the user in the validation process.

In the following, we will briefly explain the sequence of events pictured in this Kripke structure: (i) in the first moment (PastWorld2), there is a child (*Alex*) and an organization; (ii) in the second moment (PastWorld1), *Alex* studies in that organization, which plays the role of a school, and there is a second organization; (iii) in the third moment (PastWorld0), *Alex* becomes a teenager, still studying in the same school, but now also an employee (trainee) of the second organization, which plays the role of a hiring organization; (iv) in a counterfactual moment just after PastWorld0, *Alex* has undergone a heart transplant and becomes a healthy adult who works for the same organization; he no longer studies; (v) in the current moment (CurrentWorld), *Alex* is dead; (vi) FutureWorld1 depicts the possibility of both organizations continuing to exist in the future, while (vii) FutureWorld0 depicts the possibility that one of them no longer exists.

This Kripke structure exemplifies some important constraints like the rigidity of the Kind Person exemplified by *Alex* (he never ceases to be an instance of Person while he exists); the anti-rigidity of the Phases Child, Teenager and Adult (for every world w in which *Alex* is in one of these Phases, there is a world w`, accessible from $w$, in which *Alex* is not in that Phase); the anti-rigidity of the Roles Student and Employee (for every world $w$ in which *Alex* plays one of these Roles, there is a world $w$`, accessible from w, in which *Alex* does not play that Role); the relational dependence of the Roles Student and Employee (*Alex* can only play these Roles while related to a school by an enrollment (in the case he is a student), or related to a hiring organization by a payroll (in the case he is an employee)); as well as some well known conceptual modeling primitives, such as abstractness (of Person) (instances of Person have to be instances of Man or Woman), disjointness and completeness (of Man and Woman; and Child, Teenager and Adult).

Also, this Kripke structure illustrates the existential dependence of a person to his/her brain (*Alex* never changes his brain), depicted by the "essential" tag in the relationship between Person and Brain (Fig. 2). One can notice that in the counterfactual world, *Alex* changed his heart (maybe he underwent a heart transplant that saved his life). This behaviour is totally acceptable, as *Alex* is generically dependent on the «kind» Heart.

5 Related Work

Several approaches in literature aim at assessing whether conceptual models comply with their intended conceptualizations. Although many approaches (*e.g.*, [6] and [7]) focus on analysis of behavioural UML models, we are primarily concerned with structural models and thus refrain further analysis of behavioural-focused work.

A prominent example is the USE (UML Specification Environment) tool proposed in [8]. The tool is able to indicate whether instances of a UML class diagram respect constraints specified in the model through OCL. Differently

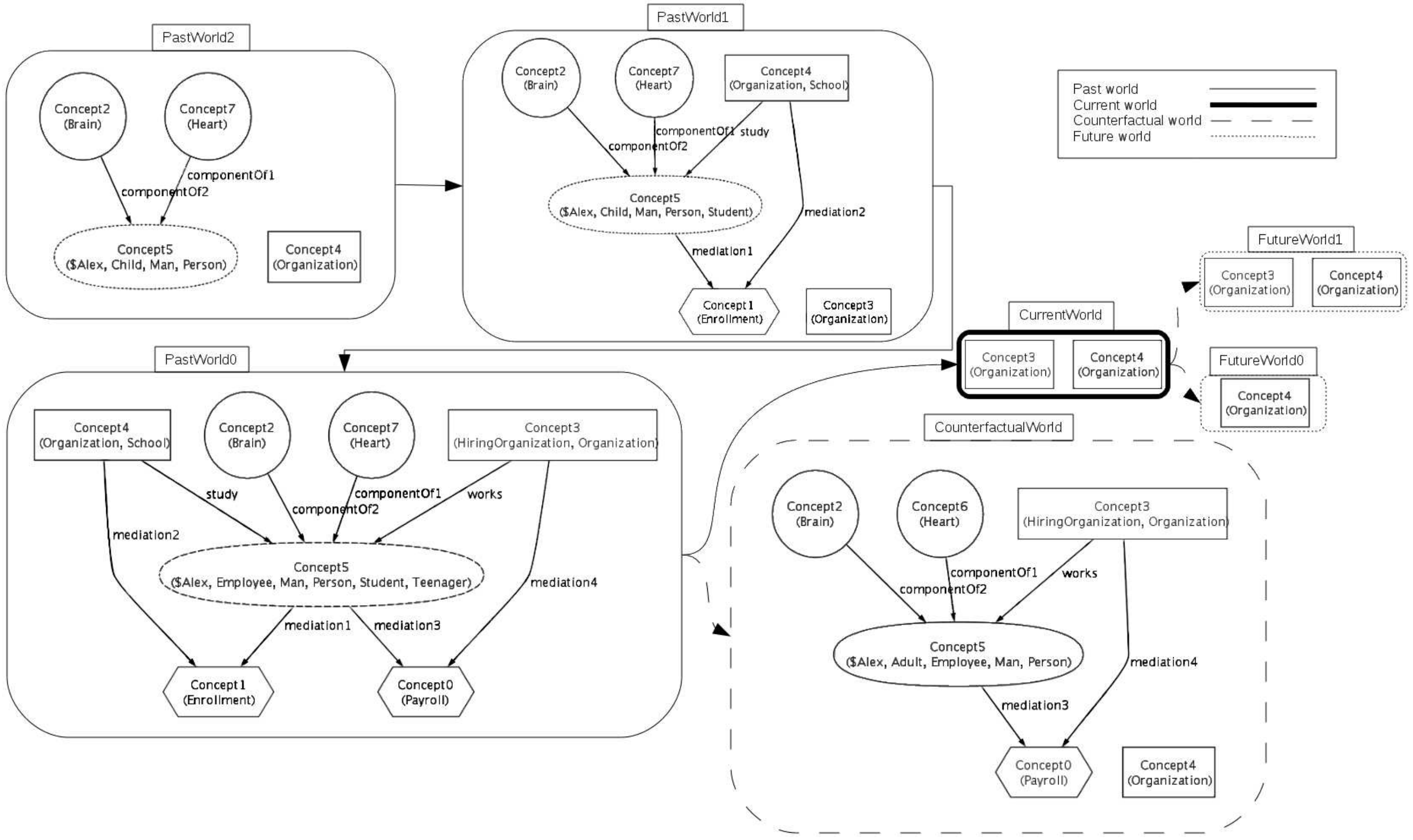

Fig. 3. Visual representation of an example world structure for the OntoUML conceptual model pictured in Fig. 2

from our approach, which is based on the automatic creation of example world structures, in USE the modeler must specify sequences of snapshots in order to gain confidence on the quality of the model (either through the user interface or by specifying sequences of snapshots in a tool-specific language called ASSL, A Snapshot Sequence Language). Since no modal meta-property of classifiers is present in UML, this tool does not address modal aspects and validates constraints considering only a sole snapshot.

Finally, the approaches of [9] and [10] are similar to ours in that they translate UML class diagrams to Alloy. However, both of them translate all classes into Alloy signatures, which suggests that no dynamic classification is possible in these approaches. Similarly to our approach, [10] implements a model transformation using model-driven techniques to automatically generate Alloy specifications, while [9] relies on manual translation to Alloy. Similar to USE, [9] focuses on analysis and constraint validation on single snapshots. [10] introduces a notion of state transition but still does not address the modal aspects of classes since these are not part of UML.

6 Final Considerations

A mature approach to conceptual modeling requires modelers to gain confidence on the quality of the models they produce, assessing whether these models express as accurately as possible an intended conceptualization. This paper contributes to that goal, by providing tools to validate the modal properties of a conceptual model in OntoUML.

Following a model-driven approach, we have defined and automated a transformation of OntoUML models into Alloy specifications. The generated Alloy specifications are fed into the Alloy Analyzer to create world structures that show the possible dynamics of object creation, classification, association and destruction as defined in the model. The snapshots in this world structure confront a modeler with states-of-affairs that are deemed admissible by the model. This enables modelers to detect unintended states-of-affairs and take the proper measures to rectify the model. We believe that the example world structures support a modeler in the validation process, especially since it reveals how state-of-affairs evolve in time and how they may eventually evolve (revealing alternative scenarios implied by the model.)

The generated Alloy specification is correct by construction such that it reflects UFO's modal axioms. As a consequence, any world structure created by the Alloy Analyzer respects UFO's modal axioms and shows that the model is satisfiable. If the Alloy Analyzer fails to find an example world structure, this may indicate unsatisfiability, although no guarantee of unsatisfiability is given. This is a consequence of Alloy's choices to cope with tractability. For instance, Alloy searches for example structures within a restricted context, *i.e.*, a given finite maximum number of elements.

As future work, we intend to incorporate support for domain constraints in our approach, *e.g.*, including OCL constraints in an OntoUML model. This will require transforming these constraints into Alloy in order to guarantee that the constraints are satisfied in all instances generated by the Analyzer.

Further, we intend to work on methodological support for the validation process, proposing guidelines for modelers to select relevant world structures. We will aim for an interactive approach in which a modeler can select which of the alternative scenarios to consider. We believe that this may help pruning the branches in the world structure keeping the size of this structure manageable.

Ideally, by exploring visualization techniques, we could use the instances generated by Alloy as example scenarios to be exposed to the stakeholders of the conceptual model (such as domain experts) in order to validate whether their conceptualization has been captured accurately by the modeler.

Acknowledgments This work has been supported by CNPq, FAPES (INFRA-MODELA) and FACITEC (MODELA). The authors also thank Kyriakos Anastasakis for his support and suggestions.

References

1. Mylopoulos, J.: Conceptual Modeling and Telos. In: Conceptual Modeling, Databases, and CASE: An Integrated View of Information Systems Development. Wiley (1992) 1
2. Guizzardi, G.: Ontological foundations for structural conceptual models. PhD thesis, University of Twente, Enschede, The Netherlands, Enschede (October 2005) 1, 2, 3, 4, 5
3. Benevides, A.B., Guizzardi, G.: A model-based tool for conceptual modeling and domain ontology engineering in ontouml. In Filipe, J., Cordeiro, J., eds.: ICEIS. Volume 24 of Lecture Notes in Business Information Processing., Springer (2009) 528–538 2
4. Jackson, D.: Software abstractions : logic, language, and analysis. MIT Press, Cambridge, Massachusetts, London, England (April 2006) 2
5. Hughes, G.E., Cresswell, M.J.: A Companion to Modal Logic. Routledge and Kegan Paul, London (January 1985) 2
6. Beato, M.E., Barrio-Solórzano, M., Cuesta, C.E.: UML automatic verification tool (TABU). In: SAVCBS'04 Specification and Verification of Component-Based Systems at ACM SIGSOFT 2004/FSE-12. (2004) 7
7. Schinz, I., Toben, T., Mrugalla, C., Westphal, B.: The rhapsody uml verification environment. In: SEFM '04: Proceedings of the Software Engineering and Formal Methods, Second International Conference, Washington, DC, USA, IEEE Computer Society (2004) 174–183 7
8. Gogolla, M., Büttner, F., Richters, M.: Use: A uml-based specification environment for validating uml and ocl. Science of Computer Programming **69** (2007) 27–34 7
9. Massoni, T., Gheyi, R., Borba, P.: A uml class diagram analyzer. In: 3rd International Workshop on Critical Systems Development with UML, affiliated with 7th UML Conference. (2004) 143–153 9
10. Maintainers: UML2Alloy. Project website: http://www.cs.bham.ac.uk/~bxb/UML2Alloy 9