Advanced Features for Model Visualization in the UML and OCL Tool USE

Martin Gogolla[1] Haleh Havakili[2] Carsten Schipke[3]

Abstract: The tool USE (UML-based Specification Environment) offers functionalities for UML model validation, verification and exploration. The tool supports to describe system structure and behavior with class models, protocol state machines and imperative operation implementations on the modeling level, utilizing OCL constraints (a) for class and state invariants as well as (b) for operation and transition contracts. Validation and verification relies on the construction of object model snapshots. This contribution focuses on advanced visualization features that have recently been developed for object models. Using these new features, it is now feasible (a) to take advantage of various automatic layout options for object models and (b) to express advanced views on object models concentrating not only on classes but utilizing also associations.

Keywords: UML; Class model; Object model; Object model visualization features

1 Introduction

The tool USE (UML-based Specification Environment) [US20] offers essential functionalities for model validation, verification and exploration of UML and OCL models. The tool allows the developer to catch system structure and behavior with class models, protocol state machines and operation implementations on the modeling level, utilizing OCL expressions (a) for class and state invariants, (b) for operation and transition contracts and (c) for imperative operation implementation on the model level [GBR07, GHD18]. Recently, USE has added new options for object models, i.e., object diagrams, that comprise (a) automatic layout options and (b) advanced views based on associations, in addition to views based on classes available formerly.

The automatic layout options allow the developer to choose from various basic layout forms (horizontal, vertical, swimlane) with variations (e.g., top to bottom, left to right). The automatic layout is parameterized by interactive settings for horizontal and vertical object distances. The automatic layout is applied only to objects and links visible at the time when the automatic layout is called and thus interacts with options for building views discussed below. For the layout, links are interpreted as being directed, and the direction (e.g., top to bottom) and with this the placement of objects follows the specification in the textual USE class model in which the roles from an association are declared in a particular order.

[1] University Of Bremen, Database Systems, Germany gogolla@uni-bremen.de
[2] University Of Bremen, Database Systems, Germany havakili@uni-bremen.de
[3] University Of Bremen, Database Systems, Germany schipkca@uni-bremen.de

Building views in an object model, i.e., the process of hiding objects and links and optionally successively showing them again, is now supported by utilizing associations. Links from particular associations can now systematically be hidden and displayed again, whereas formerly this was possible only for objects from particular classes. Furthermore, links of a particular association kind (compositions, aggregations, whole-part associations, association classes, reflexive associations, n-ary associations, ...) can be systematically considered and handled together. Displayed objects can be divided into important and for a particular purpose less important objects by showing the less important objects in a different color in contrast to colouring for prominent objects.

In our approach, object models play a central role within the validation and verification process, for example to show (counter) examples for suspected properties like model consistency or invariant independence. Therefore powerful features for visually catching important aspects and easy applicability of them are of crucial importance in our approach. The object model options that are now available in USE seem to us more flexible and expressive than the possibilities available in comparable approaches for object model and instantiation visualization like Alloy [Ja19], EMF2CSP [Go12] and EMF [SBP08].

2 Recent USE Extensions and Improvements for Visualization

Figure 1 shows a first USE screenshot in which the new options for *automatic object layout* are demonstrated. The class model (A) describes political parties that can participate in elections for an area that is divided into certain districts, as shown in the figure center. The voting result is recorded with an association class that (as usual in Germany) distinguishes between the number of first votes (for a candidate of a party) and second votes (for the party itself). Both Party and District objects show in derived attributes the accumulated sum of first and second votes for the Party and District, respectively. For example, the top-most Voting link-object in the object model in the left reflects that the (liberal) party 'FDP' received in the German federal state Hessen (HE) in year 2017 during the election for the Bundestag about 238.000 first votes (number of votes given in 1000 votes), 'FDP' in Germany received in total about 3.241.000 votes, and in Hessen about 3.262.000 people casted a first vote ballot. The default, clumsy object model (B) in the top middle together with the 'Object count' and 'Link count' windows (C) in the top right reveal that 118 objects are present (16 Districts=FederalStates + 6 (main) Parties + 96=16*6 Votings). From these 118 objects the ones connected to Hessen are selected through the 'Selection by OCL expression' window (D), and these objects are automatically layouted with the 'Horizontal layout' and the indicated spacing parameters (E). The result (F) is displayed in the left 'Object diagram' window. Additionally the Voting objects have been selected and are shown in light gray in order to put emphasis on the Party objects and the single District object.

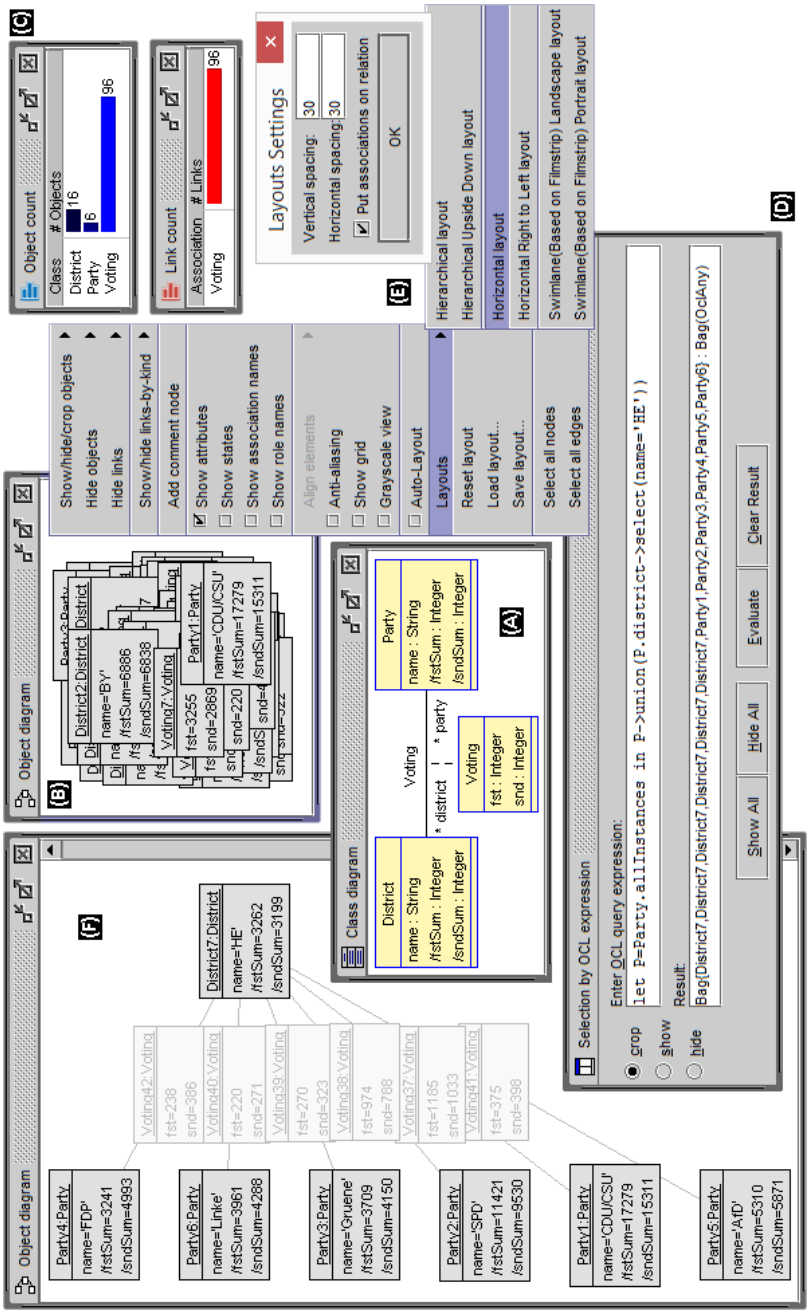

Fig. 1: New USE object model features for automatic layout and view handling (A).

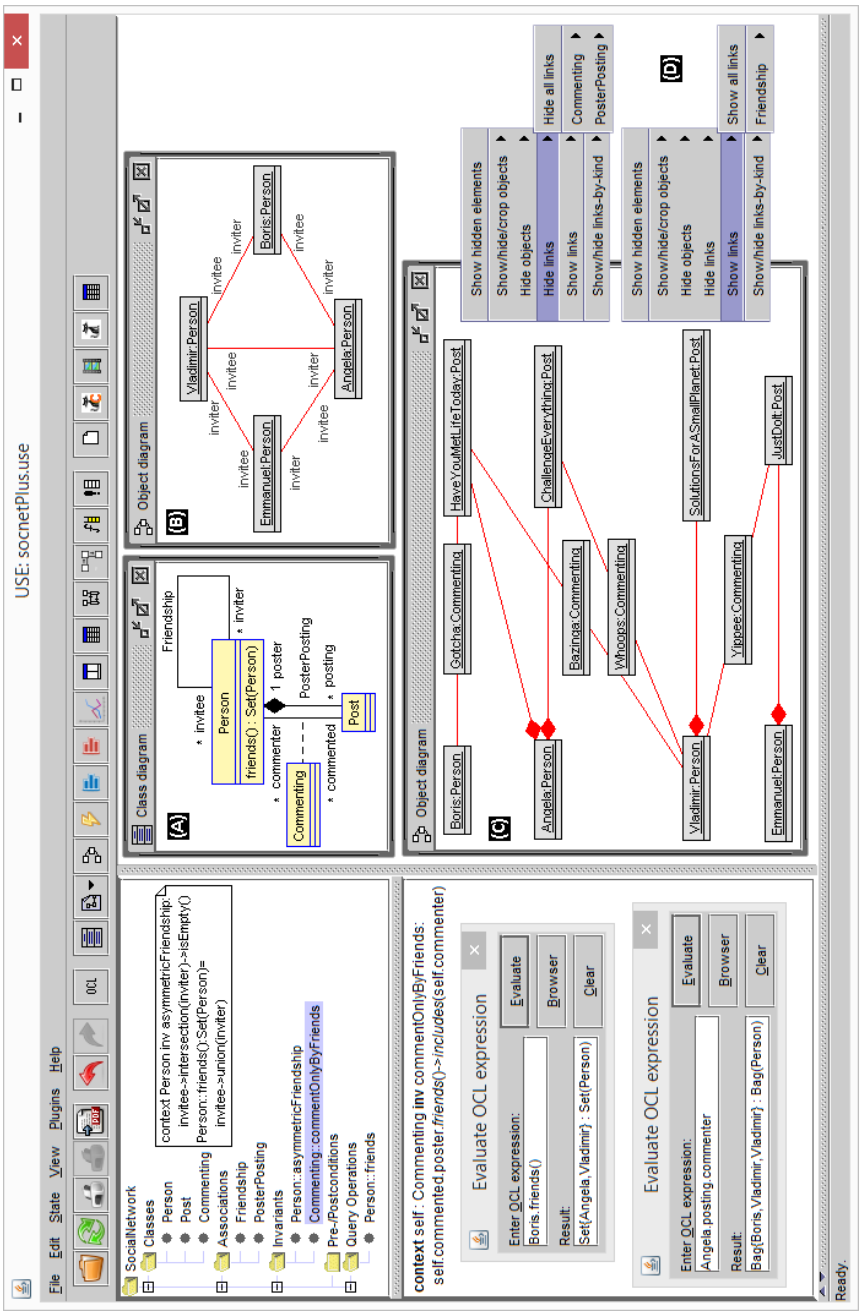

Fig. 2: New USE object model features for automatic layout and view handling (B).

Figure 2 pictures a second USE screenshot emphasizing the new options for *building object model views*. The class model (A) describes a social network with a Friendship connection between Person objects. It has features for creating and commenting postings, and two invariants restrict the allowed object models. The automatically generated object model (see [GHD18] for details on the generation concepts) is shown with two different object diagram views that focus on different aspects of the model: the Friendship aspect and the PosterPosting resp. Commenting aspect. The two different diagrams have been achieved (B) by visualizing only the Friendship links and (C) by visualizing only the PosterPosting and Commenting links; both diagrams have utilized an automatic layout whose concepts have been discussed before. The newly added USE options (D) allow the developer, for example, that in the lower object diagram the shown links (PosterPosting and Commenting) could be hidden and that the hidden links (Friendship) could be shown.

Object model usage: In order to emphasize the import role of object models in our setting, let us sketch invariant independence (each single invariant cannot be deduced from the other invariants; details see [GHD18]): As the two invariants in Fig. 2 are independent, USE supports proving independence by automatically calculating two object models (not shown here); one in which the 1st invariant holds and the 2nd invariant fails; and one in which the 1st invariant fails and the 2nd invariant holds.

3 Conclusion

We have demonstrated how the UML and OCL modeling tool USE has been extended and improved recently by new visualization features for automatic object model layout and for building sophisticated object model views utilizing associations and links.

Bibliography

[GBR07] Gogolla, M.; Büttner, F.; Richters, M.: USE: A UML-Based Specification Environment for Validating UML and OCL. Science of Computer Programming, Elsevier, 69:27–34, 2007.

[GHD18] Gogolla, M.; Hilken, F.; Doan, K.-H.: Achieving Model Quality through Model Validation, Verification and Exploration. Computer Languages, Systems and Structures, Elsevier, 54:474–511, 2018.

[Go12] González, C.A.; Büttner, F.; Clarisó, R.; Cabot, J.: EMFtoCSP: A tool for the lightweight verification of EMF models. In (Gnesi, S.; Gruner, S.; Plat, N.; Rumpe, B., eds): Proc. 1st Int. Workshop Formal Methods in Software Engineering. IEEE, pp. 44–50, 2012.

[Ja19] Jackson, D.: Alloy: A language and tool for exploring software designs. Communications of the ACM, 62(9):66–76, 2019.

[SBP08] Steinberg, D.; Budinsky, F.; Paternostro, M.: EMF: Eclipse Modeling Framework, 2nd Edition. Addison Wesley, 2008.

[US20] USETeam: USE: UML-based Specification Environment, Version 5.2. SourceForge together with University of Bremen, 2020. https://sourceforge.net/projects/useocl/.