From Graph Transformation to OCL using USE

Martin Gogolla, Fabian Büttner, and Duc-Hanh Dang

Universität Bremen, Fachbereich 3, AG Datenbanksysteme, D-28334 Bremen
http://www.db.informatik.uni-bremen.de/projects/USE/

With the tool USE, UML class diagrams[1] with additional OCL constraints can be validated and properties can be formally checked. Constraints may be class invariants and operation pre- and postconditions. USE builds system states with object diagrams[2] and expresses system evolution with operations employing basic state manipulations by creating and destroying objects (nodes) and links (edges) and by modifying attributes.[3]

A graph transformation system is expressed in USE by modeling the working graph with an object diagram and by expressing the graph transformation rules with operations modifying the working graph.[4] These operations encapsulate an executable sequence of basic state manipulations[5] and are additionally characterized by pre- and postconditions in which application conditions of the graph transformation rules can be expressed.[6] The graph transformation rules are formulated in a special language permitting to describe left and right hand side of rules as well as their application conditions. This language is automatically translated into USE command sequences and OCL pre- and postconditions. The rules may be executed interactively through operation calls.[7] The current working graph and its properties may be inspected on a graphical user interface and through evaluation of OCL expressions, e.g., for determining the rule redexes in the current working graph.[8] Additionally, OCL invariants may be used to restrict the permitted working graphs.[9]

---

[1] Figure 1 pictures in the lower third a class diagram with class `Person` having a civil status property `civstat` which is modified by events like `birth` or `marry`.

[2] Figure 1 shows an object diagram with five `Person` objects and one `Marriage` link.

[3] The class `RuleCollection` includes the operations `birth` for creating a new person and `marry` for establishing a marriage. `RuleCollection` is a singleton class.

[4] In the top, the graph transformation rule for a marriage is shown as it is specified currently. On the right, a possible graphical representation is captured. Left and right hand side of rules consist of objects, links, and OCL assertions. The rule `marry` is the origin of the operation `marry` in the class `RuleCollection`.

[5] A call of `marry` will introduce a `Marriage` link and will modify the participating `civstat` attributes as described in the rule `marry`.

[6] The precondition of `marry` will assert that only unmarried living people with the correct gender can be involved as specified in the rule `marry`.

[7] The sequence diagram shows the executed operations leading from the empty working graph to the shown object diagram.

[8] The two OCL expressions compute the names of unmarried people and the redexes of rule `marry`, i.e., the possible meaningful substitutions for the `marry` parameters.

[9] As shown in the class invariants window, all invariants are valid in the reached system state, for example, the displayed invariant `femaleNoWife`.

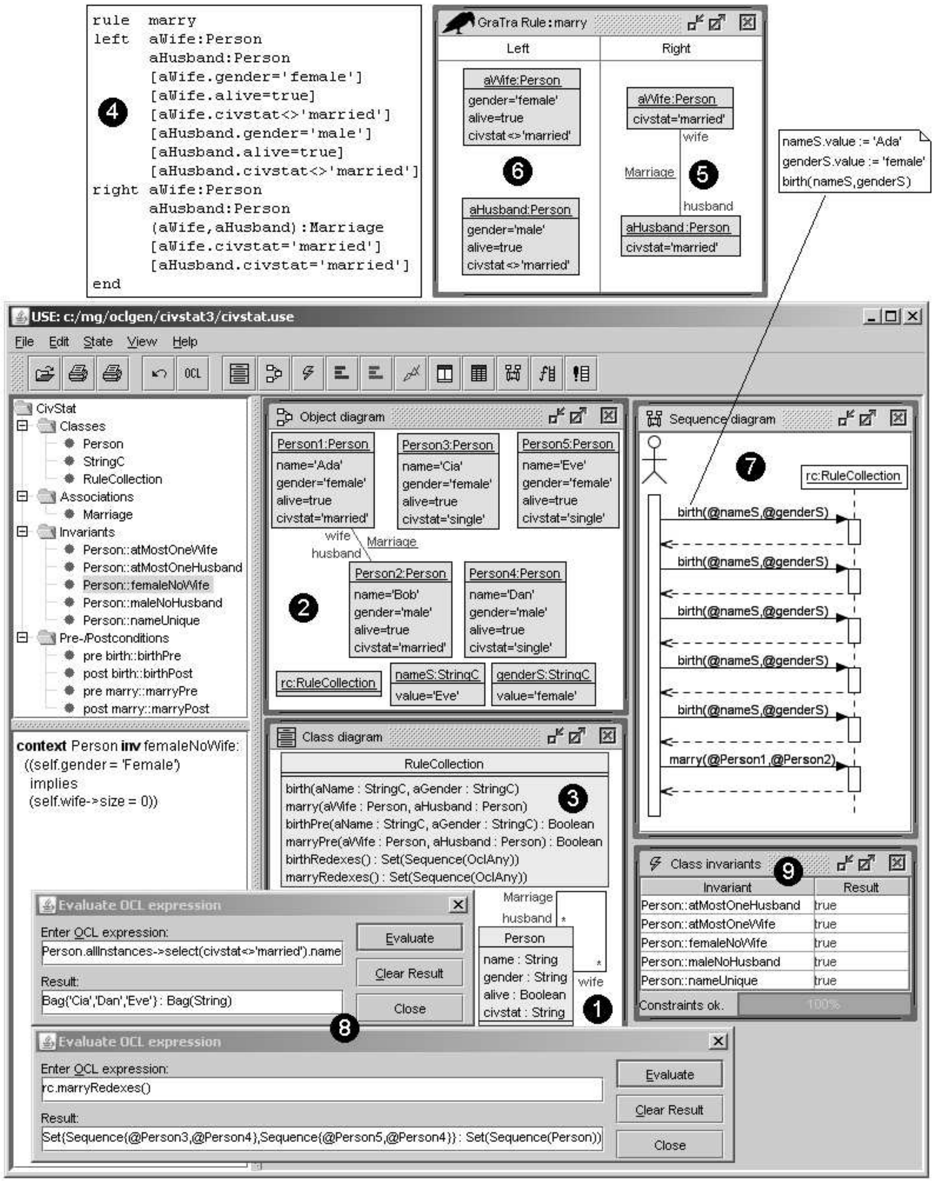

Fig. 1. Example Graph Transformation and USE Screenshot

References

[1] USE Team. USE: A UML-based Specification Environment. `http://www.db.informatik.uni-bremen.de/projects/USE/`, 2008.

[2] F. Büttner and M. Gogolla. Realizing Graph Transformations by Pre- and Postconditions and Command Sequences. Proc. 3rd Int. Conf. Graph Transformations (ICGT'2006), pp. 398-412. LNCS 4178, Springer, Berlin, 2006.