An Eclipse Plugin for Validating Names in UML Conceptual Schemas

David Aguilera, Raúl García-Ranea, Cristina Gómez, and Antoni Olivé

Department of Service and Information System Engineering
BarcelonaTech – Universitat Politècnica de Catalunya
Barcelona, Spain
{daguilera,cristina,olive}@essi.upc.edu,
raul.garcia-ranea@est.fib.upc.edu

Abstract. Many authors agree on the importance of choosing good names for conceptual schema elements. Several proposals of naming guidelines are available in the literature, but the support offered by current CASE tools is very limited and, in many cases, insufficient. In this demonstration we present an Eclipse plugin that implements a specific proposal of naming guidelines. The implemented proposal provides a guideline for every kind of named element in UML. By using this plugin, the modelers can automatically check whether the names they gave to UML elements are grammatically correct and generate a verbalization that can be analysed by domain experts.

Keywords: Naming Guidelines, Eclipse, Conceptual Schemas.

1 Introduction

Names play a very important role on the understandability of a conceptual schema. Many authors agree that choosing good names for schema elements make conceptual schemas easier to understand for requirements engineers, conceptual modelers, system developers and users [5,6].

Choosing good names is one of the most complicated activities related to conceptual modeling [8, p.46]. There have been several proposals of naming guidelines for some conceptual schema elements in the literature [3,7] but, as far as we know, few CASE tools support this activity. One example is [2], which controls that the capitalization of some elements is "correct", like "classes should start with a capital letter".

In this demonstration, we present an Eclipse plugin that adds naming validation capabilities to the UML2Tools framework. This plugin can assist modelers during the naming validation process of named elements in UML, following the complete naming guidelines presented in [1].

2 Overview of the Naming Guidelines

There are several naming guidelines available in the literature on how to name conceptual schema elements. We implemented the proposal presented in [1]

O. De Troyer et al. (Eds.): ER 2011 Workshops, LNCS 6999, pp. 323–327, 2011.

because it is complete: for each kind of element to which a modeler may give a name in UML, it provides a guideline on how to name it. As an example, two guidelines are summarized in the following:

Guideline for Entity Types

$\mathcal{G}_{1f}$ The name of an entity type should be a noun phrase whose head is a countable noun in singular form. The name should be written in the Pascal case.

$\mathcal{G}_{1s}$ If N is the name of an entity type, then the following sentence must be grammatically well-formed and semantically meaningful:

An instance of this entity type is [a|an] *lower*[1](N)

Guideline for Boolean Attributes

$\mathcal{G}_{2f}$ The name A should be a verb phrase in third-person singular number, in the Camel case.

$\mathcal{G}_{2s}$ The following sentence must be grammatically well-formed and semantically meaningful:

[A|An] *lower*(E) *lower*(*withOrNeg*[2](A)) [, or it may be unknown].

where the last optional fragment is included only if *min* is equal to zero.

As stated in [1], a name given by a conceptual modeler complies with the guideline if: a) it has the corresponding grammatical form $\mathcal{G}_f$, and b) the sentence generated from the pattern sentence $\mathcal{G}_s$ and the given name is grammatically well-formed and semantically meaningful.

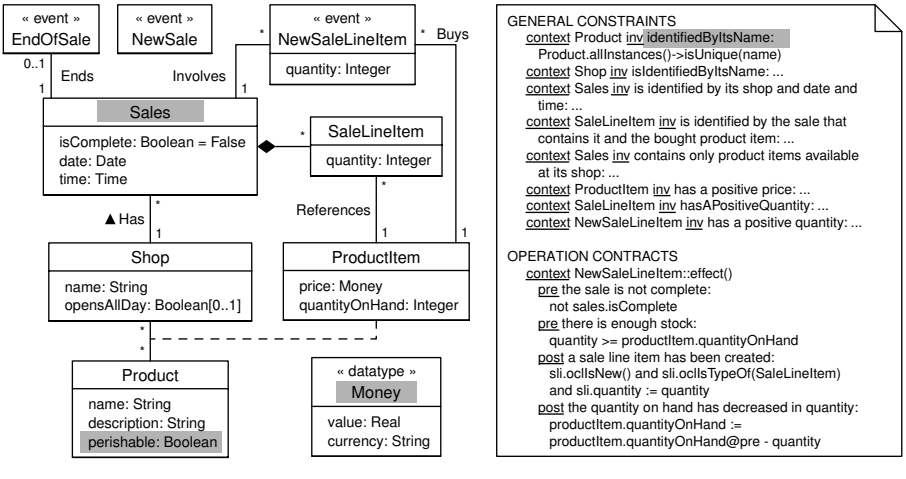

Fig. 1. Example of a conceptual schema with some names violating their guidelines

[1] *lower*(N) is a function that gives N in lower case and using blanks as delimiters.
[2] *withOrNeg*(A) extends A with the insertion of the negative form of the verb of A.

Figure 1 shows an example where some names, which are highlighted, violate their naming guidelines. The next section describes how to use the developed Eclipse plugin to detect those errors.

3 Naming Validation Plugin for Eclipse

Eclipse is an open, extensible development environment (IDE) written in Java. It is a small kernel with a plugin loader surrounded by hundreds of plugins [4]. Eclipse, in combination with the UML2Tools plugin, can manage UML files and permits modelers to define conceptual schemas.

We conceived our tool as an Eclipse plugin that extends the UML2Tools framework by adding two main functionalities. The first one checks if the names of the schema follow the grammatical form defined in their corresponding guidelines. The second functionality verbalizes the schema in a document.

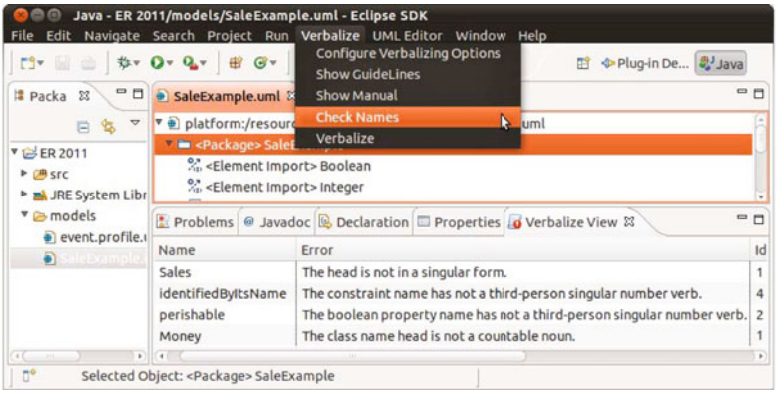

Fig. 2. Screenshot of Eclipse showing those names that violate their guidelines

Figure 2 shows a screenshot of the first functionality in action. By selecting the Package and then clicking on menu `Verbalize` ▷ `Check Names`, our tool checks the whole conceptual schema looking for errors. If the modeler selected a few elements instead of the package, only those elements are checked instead of

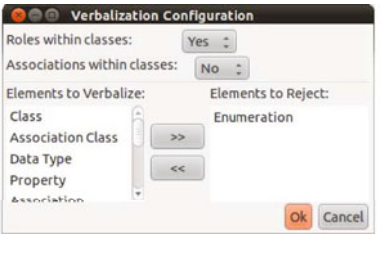

Fig. 3. Screenshot of the configuration window

the whole schema. If one or more names are incorrect, the errors are shown in a new Eclipse view.

The second functionality introduces schema verbalization. Our tool generates a PDF file containing the pattern sentences defined in the naming guidelines and the names of the selected elements. Some aspects of the resulting document can be configured by using the configuration window shown in Fig. 3. In order to generate the document, the modeler has to click on `Verbalize` ▷ `Verbalize`. Figure 4 shows the verbalization of the schema of Fig. 1 after correcting the errors previously detected. Then, the modeler or a domain expert may check the document and detect whether the generated sentences are grammatically well-formed and semantically meaningful.

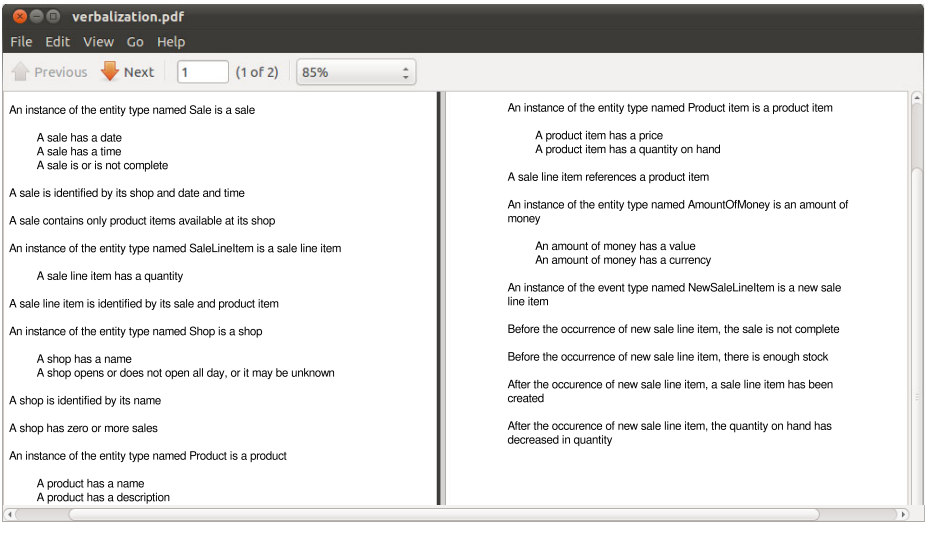

Fig. 4. Screenshot of the resulting document with the schema verbalization

Acknowledgements. Our thanks to the people in the GMC research group. This work has been partly supported by the *Ministerio de Ciencia y Tecnología* under TIN2008-00444 project, *Grupo Consolidado*, and by BarcelonaTech – *Universitat Politècnica de Catalunya*, under FPI-UPC program.

References

1. Aguilera, D., Gómez, C., Olivé, A.: A complete set of guidelines for naming UML conceptual schema elements (submitted for publication, 2011)
2. ArgoUML: ArgoUML, `http://argouml.tigris.org`
3. Chen, P.: English sentence structure and entity-relationship diagrams. Inf. Sci. 29(2-3), 127–149 (1983)
4. Clayberg, E., Rubel, D.: Eclipse Plug-ins. Addison-Wesley, Reading (2008)
5. Deissenboeck, F., Pizka, M.: Concise and consistent naming. Softw. Qual. Control 14, 261–282 (2006)

6. Meyer, B.: Reusable Software: the Base object-oriented component libraries. Prentice-Hall, Englewood Cliffs (1994)
7. Meziane, F., Athanasakis, N., Ananiadou, S.: Generating natural language specifications from UML class diagrams. Requir. Eng. 13(1), 1–18 (2008)
8. Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W.: Object-oriented modeling and design. Prentice-Hall, Englewood Cliffs (1991)