Properties of Stereotypes from the Perspective of Their Role in Designs

Miroslaw Staron, Ludwik Kuzniarz

Department of Systems and Software Engineering
School of Engineering
Blekinge Institute of Technology
Ronneby, Sweden
(miroslaw.staron, ludwik.kuzniarz)@bth.se

Abstract. Stereotypes in object-oriented software development can be perceived in various ways and they can be used for various purposes. As a consequence of these variations, assessing quality of stereotypes needs to be purpose-specific. In this paper we identify eight types of stereotypes and provide a set of criteria for assessing quality of stereotypes. The criteria for each type are formed by a set of properties that characterizes its stereotypes. The identified types are based on the purpose of each stereotype (its role in designs) and its expressiveness. We identified the types of stereotypes and their properties in an empirical way by investigating stereotypes from UML profiles used in industrial software development. The properties are intended to be used in our further research for developing guidelines for creating and using stereotypes in a more efficient way.

1. Introduction

Extending a set of modeling abstractions with dedicated constructs for modeling specific purposes is an important issue in using the Unified Modeling Language (UML, [1, 2]) for model-driven development. This general-purpose language is known to have limitations and its extensions can be seen as a means of overcoming some of those issues. One of the extension mechanisms in UML is the notion of stereotype. *Stereotypes* are means of branding the existing UML modeling elements with new semantics and properties. The notion of stereotypes, however, was introduced into object-oriented software development before the creation of UML and MDA when the stereotypes were used in a different manner than in UML.

In order to properly create and use stereotypes, modelers should be able to assess whether their stereotypes are appropriate for the purpose – i.e. *assess the quality* of the stereotypes. Thus, we perceive the quality of stereotypes from one dimension which is fitness for the purpose. In order to assess the quality we elaborate the properties which the good stereotypes should possess. The criteria for quality assessment of stereotypes can created based on finding common properties of existing stereotypes which are known to be appropriate for their purposes (an alternative way is to arbitrarily set criteria for assessing the quality). The purposes, which were identified in our previous studies, are organized into categories according to the roles

L. Briand and C. Williams (Eds.): MoDELS 2005, LNCS 3713, pp. 201-216, 2005.

of stereotypes [3]. The identified properties of stereotypes can be used within a proposed lightweight process for assessing quality of stereotypes which is presented in Fig 1. The process is based on finding common properties of a set of reference stereotypes that are known to be well suited for their purpose. The stereotypes are presented as S_R. The criteria identified, based on investigation of S_R, are used to assess quality of other stereotypes (assessed stereotypes – S_A) in the process. In order to assess the quality of S_A they must be classified (outlined in Sect. 7). The outcome of quality assessment process is an assertion whether the stereotype is good (i.e. appropriate for its purpose) or not.

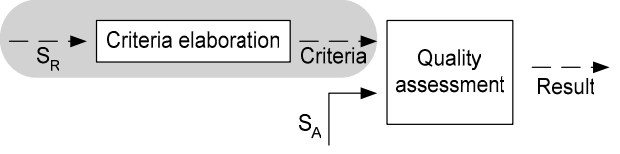

Fig. 1. A process of assessing quality of stereotypes

In this paper we focus on the way in which a set of criteria for assessing quality of stereotypes can be identified by extracting desired properties of good stereotypes (shadowed part in Fig. 1) hence we address the following research question:

> *How to elaborate quality assessment criteria for new stereotypes based on existing stereotypes which are known to be "good"?*

In this research question we identify the need for quality assessment criteria to be based on existing good practices of creating stereotypes. In our approach the identification of the properties, and their subsequent analysis, is done in an empirical way by investigating a set of 98 stereotypes used in practice. These stereotypes are grouped into profiles, which are standardized or used in companies developing UML tools and realizing the vision of model-driven software development. In the criteria elaboration we group stereotypes using categories from two classifications of stereotypes. Initially three classifications of stereotypes were considered: (i) according to their role, (ii) according to their usage scenarios [4, 5], and (iii) according to their expressiveness [6]. Based on our experiment it was found that only two classifications should be considered: (i) and (iii). Based on the results of classifying stereotypes we elaborated types of stereotypes. The *type* of a stereotype is a pair which consists of two categories to which the stereotype is classified – (C_R, C_E); where C_E is a category in classification according to expressiveness and C_R is a category in classification according to role. After elaborating the types we examined the stereotypes of each type thoroughly to identify common properties of stereotypes of each type. The properties of stereotypes we consider subsets of the following:

- kinds of data types of tag definitions (these kinds can be either data types defined in the UML metamodel or custom defined),
- kinds of the base classes the stereotype extends (concrete of abstract classes),
- kind of constraints the stereotype have,
- kind of concrete syntax (icons or guillements), and
- what kind of abstraction the stereotype should represent.

The first four properties are used to characterize stereotypes of each type. The last property is used to differentiate between types and is the basis for developing guidelines on how to choose a type of stereotype appropriate for the purpose under consideration. The guidelines are the core of our current work.

The outline of the paper is as follows. The most relevant related work in the field is presented in Sect. 2. In Sect. 3 we outline the evolution of the notion of stereotype in object-orientation and in Sect. 4 we describe the classifications used in our analysis. The design of the empirical investigation of stereotypes is presented in Sect. 5 which contains the identification of types of stereotypes. In Sect. 6 we present the properties of identified types of stereotypes. Sect. 7 suggests how the properties are to be used to assess the quality of stereotypes and Sect. 8 contains conclusions and outlines our further work.

2. Related Work

Stereotypes have been given a special attention together with the idea of the Model Driven Architecture (MDA, [7]) which is gaining popularity. The idea of models being main assets in modern software development strives for more precise models and more flexible languages to create them. As stereotypes are the main extension mechanism providing some flexibility for UML, they were evaluated in several ways by analyzing different ways of using and defining them (e.g. [8-11]). Although the stereotypes are found to be very suitable for lightweight language customization, none of the analyses performs a formal empirical study on stereotypes, which is a presented in our research.

An alternative to using stereotypes for customizing UML is to extend the metamodel of UML – i.e. to facilitate the technique of metamodeling. There exists an extension of the classification of stereotypes according to expressiveness. It is a classification of various kinds of metamodel changes developed in [12] which attempts to classify the metamodel extensions into two categories: regular metamodel extensions and restrictive metamodel extensions. The classification can be used if one wants to extend our study from stereotypes to metamodel extensions.

The analysis methods for the auxiliary study presented in our paper are based on the analysis methods used in [13]. One of the results from the study (on defect classification) is that the poor agreement between classifiers can be caused by the fact that the classifiers are not the creators of the classified objects. To some extent this claim is valid in case of our study.

An empirical approach to verification of a small-scale classification schemes in the context of requirement engineering has been done in [14]. One of the outcomes of that study is that the classification result depends on the role of the classifier and that even classifiers that are well into the domain of the built system need a considerable time in order to classify a single requirement. Furthermore, a considerable amount of time was required for getting insight into the understanding of the differences between classifiers. The design of our study also used the same means of getting insight of the classification process performed by each subject in the auxiliary study.

3. Stereotypes in UML

The idea of stereotyping was first introduced into software development by Rebecca Wirfs-Brock, who used the concept of stereotype to classify objects according to their "modus operandi" [15, 16]. Wirfs-Brock's original intention behind the usage of stereotypes was similar to the aforementioned view of stereotypes in other areas i.e. as a way of oversimplifying the view of objects' role or behavior. She used a fixed set of stereotypes, useful in characterizing the special roles of objects in the system.

An approach which is similar to Wirfs-Brock's of using stereotypes as a secondary classification of elements was adopted by the OPEN (Object-Oriented Process Environment and Notation) Modeling Language – OML [17]. Its designers perceived a stereotype as *"a facility for metaclassification"*. The initial set of stereotypes in the language was restricted (c.f. [17]), although it was divided into several groups of stereotypes (for example object, class and type stereotypes).

UML also contains a definition of stereotypes, but it specifies them as one of the possible extension mechanisms of the language. In UML, the stereotypes are a way of adding a new semantics to the existing model elements. They allow branding the existing model elements with new semantics, thus enabling them to "look" and "behave" as virtual instances of new model elements [2, 18]. They are no longer seen (at least directly in the specification) as a way of additional classification of model elements, according to their "modus operandi", but rather as a way of introducing new elements into the language, thus providing additional modeling abstractions (or providing means of adding secondary classification of the existing modeling abstractions – c.f. [9]).

During the evolution of UML (from version 1.1 [18] to 1.5 [2]), the definition of stereotypes in the UML metamodel has not changed significantly, although it underwent minor revisions due to the changes in the definition of other extension mechanisms (mostly *tagged values*, which evolved from being merely additional information for code generators in UML 1.1 specification towards virtual links between metamodel elements – tag definitions – in UML 1.3 and later). With a growing UML tool support for this mechanism the stereotypes are beginning to play a major role as a means of realizing the provision of UML as a family of languages rather than a one-fits-all modeling language [19].

In UML 2.0 [1], stereotypes are seen as a special kind of meta-classes that allow creating new modeling constructs. The constructs created in this way are intended to be as similar as possible to the original modeling constructs defined in the language specification. As far as the usage of stereotypes is concerned, the notion of stereotypes does not differ in UML 2.0 (compared with UML 1.x), but the way in which stereotypes are defined is more coherent with respect to the different levels in the four-layer metamodeling architecture.

4. Classifications of Stereotypes

Two classifications of stereotypes are considered in the course of identifying properties of "good" stereotypes. The classifications are developed independently and classify stereotypes based on distinct criteria. The classifications are summarized in

this section, while the details of them can be found in the papers where they are originally defined.

4.1. Classification of Stereotypes According to Their Expressiveness

Berner et al. [6] examined the notion of stereotype independently from object orientation within the context of modeling languages with the focus on classifying stereotypes. Their work introduced a classification of stereotypes according to the expressiveness of the stereotype, i.e. according to the amount of changes in syntax and semantics they introduce to the base model element. In their work the authors distinguished between four categories of stereotypes (denoted as C_E while defining the type of stereotype):

1. *Decorative stereotypes*, i.e. stereotypes which do not change the semantics of a language element, but change its concrete syntax (graphical representation),
2. *Descriptive stereotypes*, i.e. stereotypes which modify the abstract syntax of a language element and define the pragmatics of the newly introduced element without changing the semantics,
3. *Restrictive stereotypes* are descriptive stereotypes which modify the semantics of a language element,
4. *Redefining stereotypes*, which redefine a language element by changing its original semantics, w. r. t. syntax, they are similar to the restrictive stereotypes.

The classification attempts to address the complexity of a stereotype definition and provides guidelines for applying the different kinds of stereotypes. Although the classification addresses the problems of how stereotypes change the extended model element, it seems to neglect the problems of practical aspects of mechanisms for supporting stereotypes and the metamodeling levels that the stereotype definition concerns.

The classification aims in answering the question: "What changes does the stereotype make to the base model element?"

4.2. Classification of Stereotypes According to Their Role

It is not always the case that the classifications presented above categorize the role of a stereotype in modeling. Therefore there is a need for a classification of stereotypes based on the usage of stereotypes in software development thus capturing this role. We use the classification to categorize the notion of stereotype within the context of practical usage and introduction of stereotypes into software development, especially customizing UML tools. The classification organizes the stereotypes into three categories (denoted as C_R while defining the type of stereotype):

1. *Code Generation stereotypes*. They are aimed at making code generation rules for specific programming languages more precise and detailed, e.g. [20], intended to provide abstractions from a target programming language in order to model software using the "vocabulary" of the target programming language. Specific code generators are usually created together with the specific sets of stereotypes for code generation.

2. *Virtual Metamodel Extension stereotypes*. They are used to extend the set of UML modeling elements and perhaps to create a new "dialect" of UML, e.g. [21], intended to provide abstractions denoting new modeling constructs which are not present in the standard UML. For example these stereotypes can be used to add a "vocabulary" from another notation into UML (e.g. SPEM Profile, [22]).
3. *Model Simplification stereotypes*. They are used as an "oversimplification" of modeling elements (e.g. denoting the role of the stereotyped model element in the design), e.g. [16], intended to be created by individual modelers in an informal way. The majority of these stereotypes are used to distinguish between elements – denoting a specific purpose of the element or its role.

The detailed description of the categories is presented in [3]. The classification is based on the use of stereotypes in software development and in particular the purpose for which they are used in UML models. The proper categorization of a stereotype in this classification allows choosing the proper way of using the stereotype in UML tools.

The classification aids in answering the question – "What is the role of the stereotype in the design?"

5. Investigation of Profiles

A previous step that we conducted of identifying the properties of stereotypes was a study on comparing the classifications of stereotypes. In that study we have found that only two classifications can be used for the purpose of evaluation of quality: (i) classification according to expressiveness, and (ii) classification according to role. The third classification (the classification according to usage scenarios [4, 5]) is not considered in our study as it was found that all stereotypes in the studied profiles were categorized into one category only – i.e. type classification category.

In this paper, the set of investigated stereotypes is extended to 98 stereotypes (compared to 68 stereotypes in the previous study). The stereotypes are part of established and standardized profiles by Object Management Group (OMG): UML Profile for Software Development Processes [2, pp. 4-3 to 4-9], UML Profile for CORBA [23]; UML Testing Profile. Furthermore, we investigated profiles developed by companies and used by them: JNX profiles used by a company which developed an MDA framework [20, 24]; and profiles available in a UML 2.0 tool (Telelogic Tau [25]): TTDApplicationBuilder profile, TTDExport profile, and TTDCppAppGeneration profile. The fact that the profiles are standardized and used in industry allows using them as a set of reference stereotypes in the lightweight process of assessing quality of stereotypes.

5.1. Operation of the Study

The operation of the study is summarized in Fig. 2, where each oval represents a step taken during the study. Steps 1 and 5 are the main classification study. However, in order to increase the internal validity of the study, additional steps were required. Before classifying all stereotypes, the appropriateness of the classifier was verified by comparing the classifier's classification results to classifications of other subjects.

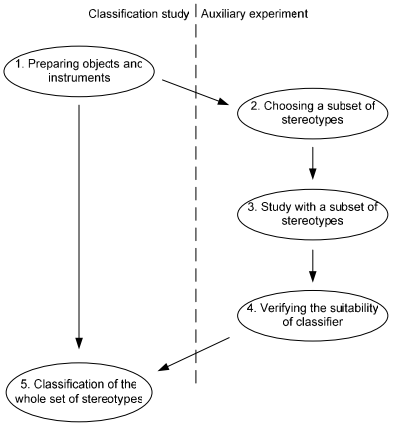

Fig. 2. Operation of the study.

Thus steps 2 through 4 are introduced as auxiliary steps used to verify the instruments before the classification study in step 5.

Using only one classifier during the study was caused by the fact the there is a substantial amount of time required to classify all stereotypes. This makes it hard to involve many subjects with appropriate knowledge for classification of all 98 stereotypes.

5.2. Auxiliary Experiment

The auxiliary study was conducted with two additional subjects. The study was performed in an academic environment with doctoral students classifying a subset of thirteen stereotypes. The subjects (and the classifier) possessed the necessary knowledge of stereotypes and they also participated in other studies on stereotypes. They were sufficiently experienced in modeling, object orientation and programming. The study afforded us with the possibility to observe whether different individuals have different understandings of the criteria used in classifying stereotypes. The results from the classifications of three subjects were analyzed with Kappa statistics [26]. The Kappa statistics measures the agreement between pairs of the variables (in this case the classification results obtained from each subject). In order to use the statistics the categories were translated to numeric values on the nominal scale. The values of the Kappa statistics are presented in Table 1 for the two subjects in the study (denoted as S1 and S2) and the classifier performing the classification on the whole set of 98 stereotypes (denoted as C).

Pair	Kappa value	Significance level	Agreement level (according to [26])
C – S1	0.24	0.0220	Fair
C – S2	0.39	0.0001	Fair
S1 – S2	0.09	0.4020	Poor

Table 1. Summary of Kappa statistics

The results indicate that there is a poor agreement (statistically non-significant – the significance level is above 0.05) between subjects S1 and S2. However, the subjects' classifications are in fair agreement with the classification done by the classifier of the whole set of stereotypes (fair agreement with both subjects, significant at the levels of 0.022 and 0.0001 respectively for C-S1 and C-S2). The fact that the subjects were in agreement with the classifier supports the decision of choosing C as the classifier for the whole set of stereotypes at the same time minimizing the risk of obtaining an incorrect classification. Nevertheless, the fair agreement level indicates that there is a dose of personal judgment in the classification. This judgment can be caused by the

fact that none of the subjects has developed these particular stereotypes that were used in the study. Furthermore, as it was also found in a similar study in [13], the fact that the subjects and the classifiers were not the creators of the objects of classification could be one of the factors that could result in the low classification agreement. In order to minimize the influence of a personal judgment, a consensus meeting was held after step 4. The objective was to discuss the different perspectives on each classified stereotype of each subject. The meeting resulted in establishing a common understanding of the different categories and which stereotypes should be included in them. Additionally the meeting provided us with the reasons for classifying particular stereotypes into each category. This in our opinion improves the objectivity of the classifier.

5.3. Results of the Study

While analyzing types of stereotypes, the starting point for considerations of the properties is the purpose of the stereotype: i.e. investigating the appropriate category in the classification according to the role of stereotypes. After the consideration of the role of stereotypes, the properties of stereotypes from the perspective of the expressiveness of stereotypes need to be considered – i.e. investigating the appropriate category in the classification according to the expressiveness of stereotypes.

The results of the study are presented in Fig. 3. The nodes in the structure represent different groups of stereotypes and the edges represent the percentage of stereotypes that belong to the appropriate category in the node below. On the lowest level of the structure there are categories in the classification according to expressiveness. In the middle level there are categories in the classification according to the role of stereotypes. The top level node represents all stereotypes in the study.

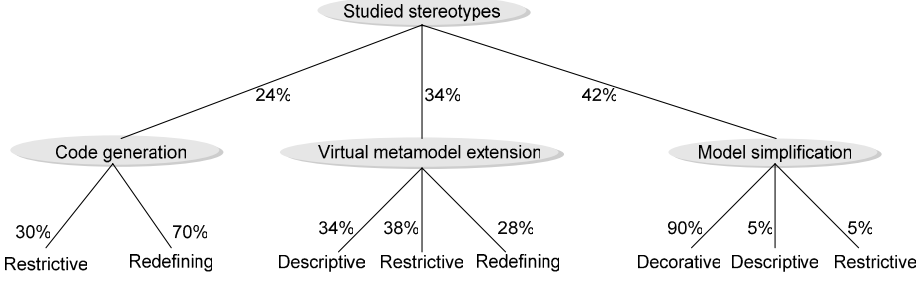

Fig. 3. Results of investigation of 98 stereotypes – grouping of stereotypes

The results of the study were the basis for elaborating types of stereotypes. Certain types of stereotypes, however, were not present in the study, e.g. (code generation, decorative). The results indicate that there exist some relationships between different categories in these classifications which make certain stereotypes (decorative) not usable for a certain purpose (code generation). The identified types of stereotypes are presented in Fig. 4 together with the percentage of stereotypes in this study that belong to each type.

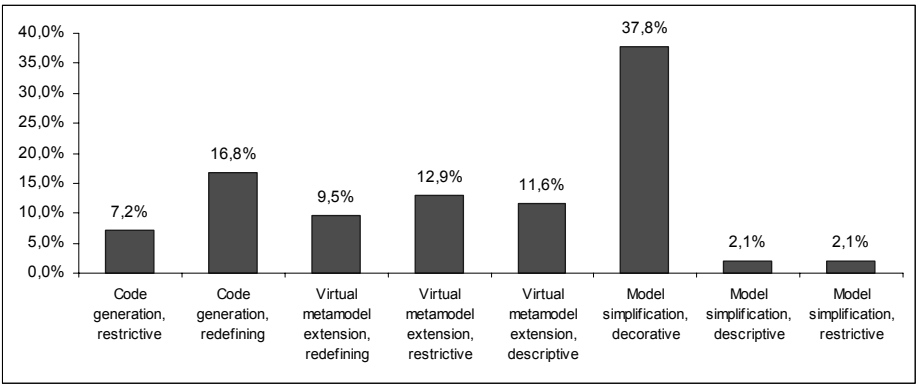

Fig. 4. Frequencies of types of stereotypes in the study

The figure shows that the most common type of stereotypes is type (model simplification, decorative). This observation shows that stereotypes are used to designate specific elements and their usage for altering the semantics of the extended elements is not very often.

6. Properties of Types of Stereotypes

We examined stereotypes of each type by conducting a qualitative analysis of their definition and usage. Properties of each type of stereotype are presented in the following section grouped according to the role of stereotypes.

6.1. Code Generation

Fig. 3 shows that there are two categories of code generation stereotypes, restrictive and redefining stereotypes. A close investigation of these two types: (code generation, restrictive) and (code generation, redefining). It shows that all code generation stereotypes possess common properties:

- base model elements (i.e. the elements the stereotype is extending) are usually concrete meta-classes from the UML metamodel as the concrete meta-classes have rules on how they should be used in the designs (some also have some standard code generation rules built into modeling tools),
- "templates" for code generation as part of the semantics of the stereotype; sometimes the "template" can be defined as a specific tagged value (provided that the code generator can interpret it), and
- no additional graphical icons defined for their presentation.

A way in which the templates for code generation are specified depends on the used code generator. The mechanisms of code generation should be investigated before creating code generation stereotypes (except, naturally, for profiles intended to be standardized – e.g. the UML Profile for CORBA – in which the templates can be

given only in textual form as the profiles are not dedicated for any specific tool – other examples of code generation templates can be found in [27, 28]).

6.1.1. Redefining Stereotypes

The stereotypes in this group redefine the semantics of their base elements. The redefinition of semantics makes the stereotyped instances of base model elements completely different from non-stereotyped ones. The semantics of the stereotypes of this type is defined by the semantics of the element in the target language which the stereotype represents. The redefinition of the semantics usually requires that the stereotyped elements are used in models only with other stereotyped elements. The redefining code generation stereotypes possess:

- tag definitions which have custom-defined data types,
- constraints restricting the usage of the stereotyped elements so that they can be used only with other stereotyped elements,
- semantics of the stereotyped elements that differs significantly from the semantics of the semantics of the model element being extended.

The data types of tag definitions are defined as part of the same profile. There exist certain exceptions as not all tag definitions have custom defined data type. A few stereotypes in the study had tag definitions which had standard data.

An example stereotype of this type is «CORBATypedef» from the UML profile for CORBA. It is a stereotype that can be applied to classes. The stereotyped classes can be used only with relation to classes that are stereotyped with other stereotypes from the CORBA profile. Furthermore, the stereotyped classes cannot have attributes and the code to be generated for the «CORBATypedef» stereotyped classes is very different than the code to be generated from non-stereotyped classes.

6.1.2. Restrictive Stereotypes

This type of stereotypes is used if an element in the target programming language is similar to an existing model element in UML although it lacks certain properties. Using model elements stereotyped with stereotypes of this type is allowed in most of the cases when the base model element can be used, but there are restrictions on the usage of the stereotyped elements(c.f. [6]). Stereotypes of this type have:

- tag definitions which types are custom-defined or built-in data types,
- constraints “restricting” the usage of the stereotyped elements in certain situations in which the base modeling element can be used,
- semantics making the semantics of the extended model element more precise.

The data types used for tag definitions can be both custom-defined (most often) or built-in.

As example stereotype of this type is «CORBAStruct» from the UML Profile for CORBA. The stereotype is applied to classes in UML and makes the semantics of classes more precise, e.g. that all the attributes should be public. Its constraints state that the «CORBAStruct» stereotyped class cannot be used in some situations when the non-stereotyped class can be used.

6.2. Virtual Metamodel Extension

Virtual metamodel extension stereotypes can be categorized into three categories in classification according to expressiveness: redefining, restrictive and descriptive. The virtual metamodel extension stereotypes are stereotypes that are used to add new constructs into UML.

6.2.1. Redefining Stereotypes

Virtual metamodel extension stereotypes which are also redefining stereotypes are stereotypes that are intended to create a new (sub-) language based on UML. In many cases these stereotypes have been created based on metaclasses from another metamodel (further referred to as the defining metamodel). The defining metamodel specifies "a language" that is intended to be used as a member of the UML family of languages. Examples of this kind of metamodel-based profiles are the SPEM Profile or the UML Testing Profile [29]. They both have defining metamodels, which are then "translated" into UML profiles. The UML Profile for CORBA is similar, but the designers of the profile explicitly name the metamodel "virtual" and use only stereotypes in it.

The stereotypes of this type have:

- constraints stating that the stereotyped elements are allowed to be used only with other stereotyped elements,
- tag definitions (if defined) of custom-defined types (corresponding to attributes of the model element in the defining metamodel which is the basis for creating the stereotype),
- semantics of stereotypes in this group differs from the semantics of their base elements and is defined based on the defining metamodel (which is the base for abstractions denoted by stereotypes) and not the UML metamodel,
- base classes are concrete meta-classes in the UML metamodel as the redefinition of the semantics is usually restricted to only the concrete meta-class being extended, and
- icons with concrete syntax specified for the defining metamodel.

Examples of this type of stereotypes are stereotypes presented in [6] as typical redefining stereotypes, «actor» and «use case» in early versions of UML. These stereotypes change the meaning of the standard modeling elements (class) and in fact create a new kind of diagrams in UML (use case diagrams) which are different from class diagrams.

6.2.2. Restrictive Stereotypes

The difference of this type in comparison to the previous type is that the semantics of the stereotypes is based on the semantics of the base model element, e.g. when the abstraction in the defining metamodel is based on an element in the UML metamodel. Just as the redefining virtual metamodel extension stereotypes they are based on defining metamodels. Thus the stereotypes in this group have:

- constraints restricting the usage of the stereotyped elements – they cannot always be used in places of the base model elements,

- base classes are most often (but not always) concrete meta-classes from the UML metamodel, the abstract meta-classes are rather uncommon as base classes for this type of stereotypes,
- tag definitions of custom-defined types (corresponding to attributes of the model element in the defining metamodel which is the basis for creating the stereotype),
- semantics making the semantics of the extended model element more precise, and
- icons with concrete syntax specified for the defining metamodel.

A representative of this type of stereotypes is «GRMdeploys» from the UML Profile for Performance, Schedulability and Time [30]. The stereotype restricts the usage of the stereotyped element in certain situations and the data type of the tag definition is custom-defined (defined as part of the same profile).

6.2.3. Descriptive Stereotypes

Virtual metamodel extension stereotypes which are descriptive stereotypes are defined in order to make the structure of existing modeling elements more precise in a new context (although these stereotypes usually do not represent elements from any defining metamodel). They are characterized by:

- types of tag definitions being usually standard data types specified in the UML metamodel,
- base classes can be both abstract and concrete meta-classes from the UML metamodel,
- no constraints restricting the usage of the stereotyped elements, and
- usually no icons.

The tag definitions are usually used by external tools. An example stereotype of this type is the «GenericExport» stereotype from the TTDExport profile in Telelogic Tau G2. The intent of this stereotype is to provide means of connecting elements in the model with other artifacts, for example relating a class to a piece of Java code in Eclipse (but not for the generation of the code itself) or linking a class to a requirement in Telelogic DOORS.

6.3. Model Simplification

Model simplification stereotypes are intended to be used for designating certain model elements. Thus they are usually the simplest of stereotypes and they merely change the concrete syntax of the stereotyped element. The name of the stereotype is usually the main element of its definition and it reveals the intention of the stereotype of all types of stereotypes for model simplification. Most of the stereotypes created for this purpose belong to the category of decorative stereotypes (90% of all model simplification stereotypes). A common property of model simplification stereotypes is that their semantics is specified in a very loose form, e.g. only an intension of a decorative model simplification stereotype.

6.3.1. Decorative Stereotypes

Most of the model simplification stereotypes are decorative stereotypes. They have the following properties:

- no tag definitions,
- no specific semantics, i.e. they are used for decoration of specific model elements, and
- icons associated with them to enable more effective recognition of stereotyped elements.

An example stereotype in this group is the stereotype «hidden» from the set of predefined stereotypes in Telelogic Tau G2. The stereotype means that the stereotyped elements should not be visible in a certain view in the tool. The stereotype does not add new properties to the extended model element, but it causes that stereotyped model elements are treated in a different way in model explorer in the tool (although they are treated in the same way as non-stereotypes model elements in models).

6.3.2. Descriptive Stereotypes

Some model simplification stereotypes add properties to the stereotyped model elements. The tag definitions provide additional information about the "context" of making the element distinct. The properties of this group of stereotypes are:

- tag definitions which usually are of standard data types,
- no constraints, and
- no icons.

An example stereotype in this category is «commentedClass» from [5, p. 155] which provide means of adding information (as a tag definition) about authors of classes in UML designs thus making the stereotyped elements special in the design.

6.3.3. Restrictive Stereotypes

Finally, there are also certain stereotypes, which to some extent restrict the usage of the stereotyped element though they are not intended to create a new modeling element (thus they are not virtual metamodel extension stereotypes). They are used to designate the elements and impose light restrictions on the elements denoting that the simplified element sometimes should not be used (the restrictions then designate the situations in which the stereotyped element should not be used). The restrictions are usually specified only informally. The properties of these stereotypes are:

- tag definitions which are usually of standard data types,
- constraints that restrict the usage of the stereotyped model element, and
- no icons.

An example stereotype which was found to be a restrictive model simplification in the study is «UseCasePackage» from the UML Profile for Software Development Processes [2, p. 4-4]. Applying the stereotype restricts the packages to be used only in specific contexts.

7. Basic Guidelines on Assessing Quality

The elaborated properties of the types of stereotypes are to be used as assessment criteria. Modelers who use the properties for assessing quality of a particular stereotype should:

1) Find the type of the stereotype, i.e. answer the questions:
   a) What is the purpose of the stereotype?
   b) What changes the stereotype introduces to the base model element?
      i) Does it redefine the semantics of the base model element (i.e. is it redefining)?
      ii) Does it make the semantics of the base model element more precise (i.e. is it restrictive)?
      iii) Does it add any tag definitions (i.e. is it descriptive)?
      iv) If none of the above, then it is a decorative stereotype.
2) Check whether the stereotype has properties of the stereotypes of this type

The stereotypes which are of a good quality should possess the properties. These well-designed stereotypes save the effort for their maintenance since their definition is as easy as it is possible given their purpose. For example the model simplification stereotypes are very simple since they are used for simple purposes while the virtual metamodel extension stereotypes are more complex since they are dedicated for more advanced purposes.

There might be other types of stereotypes than the types found in the study although we made our best efforts to include stereotypes from various vendors and for diverse purposes in order to make the study as broad as possible. If the stereotype is of a type that is not included in the study it might be the case that the stereotype is too complex for the purpose it is supposed to serve (e.g. a redefining model simplification stereotype) and therefore it should be redesigned. Sometimes it is a case that a stereotype is intended to play two roles – then the stereotype should be redesigned and split into two stereotypes. It is important that the stereotypes are "coherent" in the sense that they are serving a single purpose and they are of a single type.

8. Conclusions

Stereotypes play an important role in using UML in an effective way. The set of standard UML constructs is known to be insufficient for all purposes and the users of UML often create stereotypes to enrich their set of modeling elements. Since stereotypes are a notion which is defined by the users of the language and it is supposed to be instantiated in user models, thus being a part of the language, the quality assessment is specific. Furthermore, due to the fact that there are various reasons for which the stereotypes, these reasons influence the way in which the quality of stereotypes should be assessed. In this paper we provide a way in which a question "What is a good stereotype?" can be answered. This paper presents a part of the lightweight process for assessing quality of stereotypes and addresses the research question on how the existing stereotypes can be used for creating criteria for assessing quality of new stereotypes. It includes an investigation of a set of stereotypes used in industry aimed at identifying types of stereotypes. The types of stereotypes reflect the

purpose for which the stereotypes are created and the changes which the stereotype introduces to its base model element.

The identified properties of stereotypes are designed to be used in assessing the quality of stereotypes that have already been created. The assessment is done in the final phase of creation of stereotypes. Currently in our research we focus on developing a set of guidelines for creating stereotypes which are appropriate for their purposes. The guidelines are intended to aid modelers to create "good" stereotypes for their purposes in a structured way. The intention of the guidelines are designed to be in a form of simple questions that would guide modelers through the process of creating the stereotype, beginning from an initial idea of what the stereotype is for and ending with the set of properties which the stereotype should possess. In our further research we intend to validate the method in a company creating a framework for model-driven software development.

References

1. Object Management Group, "Unified Modeling Language Specification: Infrastructure Version 2.0", OMG, 2004, www.omg.org, last accessed 2004-02-20.
2. Object Management Group, "Unified Modeling Language Specification V. 1.5", OMG, 2003, www.omg.org, last accessed 2004-10-01.
3. Kuzniarz L. and Staron M., "On Practical Usage of Stereotypes in UML-Based Software Development", In the Proc. of Forum on Design and Specification Languages, Marseille, 2002, pp. 262-270.
4. Atkinson C., Kühne T., and Henderson-Sellers B., "Stereotypical Encounters of the Third Kind", In the Proc. of The 5th Int. Conf. on UML, Dresden, Germany, 2002, pp. 100-14.
5. Atkinson C., Kühne T., and Henderson-Sellers B., "Systematic Stereotype Usage", *Software and Systems Modeling*, vol. 2, 2003, pp. 153-163.
6. Berner S., Glinz M., and Joos S., "A Classification of Stereotypes for Object-Oriented Modeling Languages", In the Proc. of The 2nd Int. Conf. on UML, Fort Collins, CO, USA, 1999, pp. 249-64.
7. Miller J. and Mukerji J., "MDA Guide", OMG, 2003, http://www.omg.org/mda/, last accessed 2004-01-10.
8. Gogolla M. and Henderson-Sellers B., "Analysis of UML Stereotypes within the UML Metamodel", In the Proc. of The 5th Int. Conf. on UML, Dresden, Germany, 2002, pp. 84-99.
9. Atkinson C. and Kühne T., "Rearchitecting the UML Infrastructure", *ACM Trans. on Modeling and Comp. Simulation*, vol. 12, 2002, pp. 290-321.
10. Atkinson C. and Kühne T., "The Role of Metamodeling in MDA", In the Proc. of Workshop in Software Model Engineering, Dresden, Germany, 2002.
11. Atkinson C. and Kühne T., "Model-Driven Development: A Metamodeling Foundation", *IEEE Software*, vol. 20, 2003, pp. 36-41.
12. Schleicher A. and Westfechtel B., "Beyond Stereotyping: Metamodeling Approaches for the UML", In the Proc. of Hawaii Int. Conf. on Syst. Sciences, Maui, HI, USA, 2001, pp. 10-17.
13. Henningsson K., Wohlin, C., "Assuring Fault Classification Agreement - an Empirical Evaluation", *Proc.* Int. Symposium on Empirical Software Engineering, 2004, pp. 95-104.
14. Hertzum M., "Small-Scale Classification Schemes: A Field Study of Requirements Engineering", *Computer Supported Cooperative Work*, vol. 13, 2004, pp. 35-61.
15. Wirfs-Brock R., "Stereotyping: A Technique for Characterizing Objects and Their Interactions", *Object Magazine*, vol. 3, 1993, pp. 50-3.

16. Wirfs-Brock R., Wilkerson B., and Wiener L., "Responsibility-Driven Design: Adding to Your Conceptual Toolkit", *ROAD*, vol. 2, 1994, pp. 27-34.
17. Firesmith D. G., Henderson-Sellers B., and Graham I., "The Open Modeling Language (OML) Reference Manual", New York, Cambridge University Press/Sigs Books, 1998.
18. Object Management Group, "UML Specification ver. 1.1", OMG, 1997, www.omg.org, last accessed 2004-10-11.
19. Cook S., "The UML Family: Profiles, Prefaces and Packages", In the Proc. of The 3rd Int. Conf. on UML, York, UK, 2000, pp. 255-264.
20. Staron M., Kuzniarz L., and Wallin L., "A Case Study on Transformation Focused Industrial MDA Realization", In the Proc. of 3rd Workshop in Software Model Engineering, Lisbon, Portugal, 2004.
21. Evans A., Maskeri G., Sammut P., and Willians J. S., "Building Families of Languages for Model-Driven System Development", In the Proc. of 2nd Workshop in Software Model Engineering, San Francisco, CA, 2003.
22. Object Management Group, "Software Process Engineering Metamodel Specification 1.0", OMG, 2001, www.omg.org, last accessed 2004-02-01.
23. Object Management Group, "UML Profile for CORBA", OMG, 2002, www.omg.org, last accessed 2004-10-10.
24. Staron M., Kuzniarz L., and Wallin L., "Factors Determining Effective Realization of MDA in Industry", In the Proc. of 2nd Nordic Workshop on the Unified Modeling Language, Turku, Finland, 2004, pp. 79-91.
25. Telelogic, "Telelogic Tau G2", 2004, http://www.telelogic.com.
26. Altman D., "Practical Statistics for Medical Research", Chapman-Hall, 1991.
27. Kuzniarz L. and Ratajski J., "Code Generation Based on a Specific Stereotype", In the Proc. of Information Systems Modeling, Roznov, Chech Republic, 2002, pp. 119-128.
28. Sturm T., von Voss J., and Boger M., "Generating Code from UML with Velocity Templates", In the Proc. of The 5th Int. Conf. on UML, Dresden, Germany, 2002, pp. 150-161.
29. Object Management Group, "Unified Modeling Language: Testing Profile", OMG, 2004, www.omg.org, last accessed 2004-02-14.
30. Object Management Group, "UML Profile for Schedulability, Performance and Time", OMG, 2002, www.omg.org, last accessed 2003-09-20.