A Reuse Model for Formally Verified UML Diagrams

José Sáez, José Luis Fernández, Ambrosio Toval {jsaez, aleman, atoval}@dif.um.es
Grupo de Ingeniería del Software, Dpto. de Informática, Lenguajes y Sistemas
Universidad de Murcia
☎ (+34) 968 36 46 03 - Fax: (+34) 968 36 41 51
Esperanza Manso, Miguel Angel Laguna {manso, mlaguna}@infor.uva.es
Departamento de Informática - Universidad de Valladolid
☎ (+34) 983 42 30 00 - Fax (+34) 983 42 36 71
Francisco José García - fgarcia@gugu.usal.es
Departamento de Informática y Automática - Universidad de Salamanca
☎ (+34) 923 29 44 00 - Fax (+34) 923 29 45 14

Abstract

Two of the most important issues related with the reuse of software artefacts are the localisation and retrieval of these, and the reliability that they satisfy a set of required properties. Several research groups study both problems. One of the solutions for the first problem has been the construction of asset repositories. The second one has been addressed with the verification approach of certain properties inside the context provided by some formal language or technique.

This work tries to establish a process framework that allows inserting, verifying, and retrieving software assets in a trustworthy way. In a first approach, the application of this process is directed to UML diagrams, and consequently, we have the aim to move the promises of software reuse closer to real software engineering practices.

Key Words:

Analysis asset, Design asset, Software reuse, UML, Formal verification.

1. Introduction

The systematic reuse of software artefacts brings a lot of benefits: increasing productivity, work amortisation, obtaining higher trust (*because the assets have been tested in many places and in different contexts*), and so on. On the other hand, there are several difficulties, some of which are being treated by different research projects[1].

Another important issue is reliability. Nowadays, in a development environment where third parts components (*whose internal composition is unknown, or we are not interested in knowing it, due to its complexity*) reuse is a usual activity, the need to ensure the suitableness of these elements to reach a certain quality or security levels, or satisfying a set of essential properties or requirements, will grow up. In this case, the use of formal techniques to perform the asset verification is especially suitable. Like Meyer says, the extra effort of applying mathematical techniques to specify software becomes economically justifiable, when they are applied to the development of reusable components [5]. This author has recently presented a project whose main goal is to obtain a set of trusted components, using design by contract, formal methods, formal validation, and many others [6].

The Software Engineering Group (*GIS, Grupo de Ingeniería del Software*) of the University of Murcia is working in the formalisation of graphical techniques and the formal verification of diagram properties. The study is now focused on some kinds of UML diagrams: (*simplified*) class diagrams, object diagrams and statechart diagrams [1] (*the set of the considered diagrams will be extended with others UML diagrams in the future*).

The formal reference framework for diagrams verification is established by the formal language

[1] For example, in Valladolid University (*Spain*), the GIRO group (*Grupo de Investigación en Reutilización y Orientación al Objeto*) is working in the storage, localisation and retrieval of complex reuse artefacts called Mecanos [2].

OBJ and its extensions to Maude [7].

This paper describes an initial process model to insert, verify and retrieve assets from a software repository in a reliable way. The model is applied taking UML diagrams as a particular case. Therefore, in this work, two different research areas are combined, which are represented inside MENHIR by the two above-mentioned research groups. First, the asset storage, retrieval and distribution issue is being studied by GIRO group. On the other hand, the formal verification of analysis or design models properties (*created using UML*), as interest topic for the GIS group [11].

The remainder of the document is organised as follows. Section two makes a brief introduction to the GIRO repository, as asset storage medium for its possible reuse. Section three presents the verification process and the formal framework. Section four is considers the combination of the verification results with other interesting metrics for the assets, and the introduction of this information into the repository as part of the asset's qualification reuse model. Finally, section five closes the paper with a short summary, including the main work conclusions, and the description of the subsequent steps in the presented process definition.

2. The GIRO repository

The GIRO repository (*http://jupiter.dcs.fi.uva.es*) is the fundamental basis of the reuse process. It is the storage part for the reusable artefacts. This repository has several functional characteristics, being the search and the localisation the minimum basic ones. The GIRO repository has been based on EUROWARE[2] repository engine. Until now, the software artefacts obtained in the software systems development life cycle have been stored in this repository.

The digital image-processing and software for handicapped are the two main application domains inside the repository, especially the first one with object-oriented assets from different life cycle stages and built them with several CASE tools.

The initial analysis or design asset storage format has been a word processor format (*typically MS Word format or RTF*). This option is clearly insufficient for our goals. Now, and thinking in the new GIRO repository construction (*based on ORACLE, with the Mecano model [4] implemented as a data base schema*), a different approximation has been made. First, independently of the abstraction level, each asset is associated with two kinds of documents:

- Informative documents (perhaps based on HTML, thinking in the use of the repository over Internet and in the platform independence too).
- Compulsory, one document expressing the asset itself at least. This one allows the asset consulting and its incorporation into a current development. This format should allow the asset study and modification in the local computer, if the user has the asset creation tool[3].

The particular asset format is a important issue. Our proposal is directed to ensure one commercial format with a wide diffusion and standard (*CDIF, XML…*) if it were possible. In the particular case treated in this paper, the UML diagrams, we plead in favour of the practical possibilities of the Rational ROSE tool, which allows exporting parts of a complex model. For example, you can export a package, a class diagram or only an isolated class.

The practical issue establishes that the repository user has installed in its local computer a ROSE tool version (*it doesn't matter if it is a trial or a demo version*). Therefore, the internet browser can execute this application when it was required, at the same way it would be able to start the appropriate development environment when it finds a C++, Eiffel or Java file.

---

[2] *Enabling Users to Reuse Over Wide AREas* [10].

[3] This characteristic was presented in the implementation abstract level assets (*source code, easy to modify with a text editor*), but now the goal is extending this property to the other abstraction levels.

This running has been proved under MS-Windows platform and it presents an acceptable operation. A similar execution environment has not been tested under other platforms, i.e. UNIX, because there is not accessible free tools.

The interchange format is another important and interesting issue. There is a proposal to use XML as independent interchange format to express every UML diagram, including the appropriate DTD [8]. This idea is being studied for adoption for the new GIRO repository. The main problem is that XML could be only an interchange format, without graphical tool associated to each kind software artefact represented by it. This problem requires importing each file from the correct tool, and this is a not suitable situation. In this case, it would be more appropriate to develop a suite of graphical plug-in viewers for the UML diagrams.

The quality of this kind of asset is certified in a separate first-group file (in HTML format). As the qualification model says [9], the assets could be introduced by every authorised repository user, but the asset should pass the qualification process specified in the quality plan. In a first stage, the asset is classified as "*no audited*" one (*and everybody can access it*), because an auditory is the first filter for an asset.

When the asset is audited, the qualification process goes on. It's important to remark that some kinds of assets must be qualified following a manual process by the repository administrators (*business rules, user's requirements…*). Other assets could be subject to different controls based on metrics, using automatic tools if it were possible (*following the mecano's qualification model procedures*). And finally, other assets could be subject to formal verifications.

Each time that an authorised user introduces or modifies an elemental asset, an e-mail is sent to the repository administrators with the asset and the associated information. Several repository administrator categories exist (*general administrator, auditor, verifier, certificator…*). Each category can be associated to one or many people, and then a single person can play different roles inside the qualification model.

Every kind of registered asset (*UML class diagram, DFD…*) has a control-quality process (*based on the qualification reuse model*), that is defined by the general administrator. This control process includes a list of responsible peoble who have to be informed about the assets entry or the end of the initial audit process.

For example, the members of the GIS group could be the target of the assets to perform their study. This group will develop an automatic process to translate the UML diagrams to OBJ/Maude context. Other possibility is to integrate the automatic tool as repository service, this way the verify team receives the transformed diagram.

3. The component verification process model

When the diagram is received, its OBJ/Maude translation is started. Maude is an OBJ extension based on equational logic and rewriting logic. Maude supports functional modules like OBJ, systems modules for dynamic specification aspects and object-oriented modules to facilitate the definition of concurrent object-oriented systems. In the presented case, each diagram is represented as an only term built over a signature from a rewriting theory. For example, the type *Company* is represented by the **tComp** term, that is shown in Figure 1.
The storage format of the assets in the repository is very important for the translation tool from UML diagrams to OBJ/Maude (*also for using them in other tools*).

After translation, the testing set is applied (*these tests are defined in the OBJ/Maude verification framework*). The number of currently available tests is limited, because the work done until now has been directed in the definition of the correspondence between UML and OBJ/Maude models.

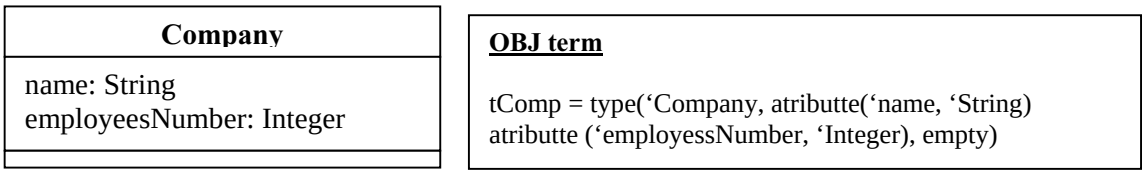

Figure 1. Company type and its corresponding OBJ term

Actually, the state orthogonality can be verified (static semantic) in a statechart. The violation of the cardinality of a binary relationship (dynamic semantic) verification in a class diagram animation (using an adequate tool) is possible, too.

Taking the statechart representation as a tree structure:

- Two states **A** y **B** are orthogonal if these states are not in the same path of the state tree and their nearest common ancestor is an AND state.
- A complex transition (or a simple one) fulfils the orthogonality constraints when all of its source states are pairwise orthogonal, all of its target states are pairwise orthogonal, and each target state is non-orthogonal with regard to each source state
- A statechart is orthogonal when all its transitions are orthogonal.

The violation of the cardinality of a binary relationship is a dynamic semantics property. This property is tested when the current system state evolves at a given time, creating new objects or deleting existing ones in the actual state of the model. It's possible to apply the verification of this property to an object diagram, that is to say, a class diagram static instance.

In the future, available test will include obtaining the derived relationships with their associated cardinalities. In any case, the important idea at the process level, is to guarantee a testing set for the UML diagrams.

The results generated by the test application to the UML diagrams could be grouped in two categories.

First, the indication of invalidation or the diagram inconsistencies results, after the application of the tests of validation. A negative result in these tests involves returning the diagram to its author to be corrected.

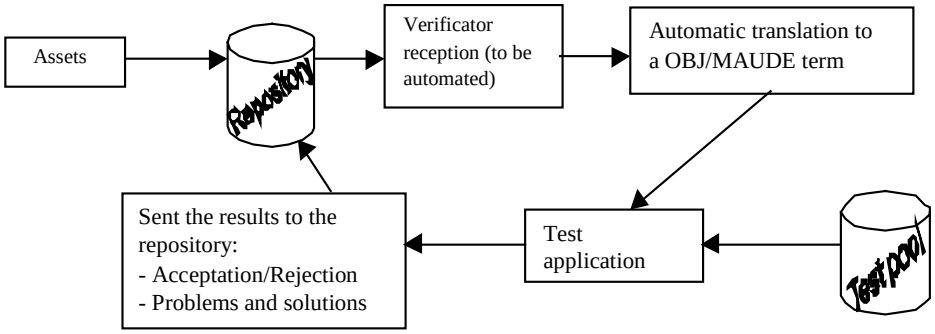

Figure 2. Verification Process.

The other test group is destined to detect some kinds of problems that don't invalidate the model, but that are conflictive design points. The results of the application of these tests are detected problem indicators and their possible solutions. These results can be added to the diagram documentation in the repository. In a first approximation, these add-ons could be some text inserted in a web page. This solution is easy to be integrated in the GIRO repository, but it has a serious

inconvenient: the results from formal test could not be used as search and retrieval criterion. When the number of the formal tests grows up, a more structured form of representation results should be defined, allowing their inclusion as search and retrieval criterion (*this must be done under the qualification reuse model for the assets in the repository, presented in Figure 2 and defined in [9]*).

4. Results interpretation and publication

The results from the verification of the properties will be inserted into the informative document set of the verified asset. Therefore, and thinking in an Internet access, the HTML format seems the most suitable one, because it offers the same portable facilities as ASCII format and adds presentation facilities.

Also, there are other certification-elements to present (*class metrics, quality factors…*)[4]. The quality responsible must add all information elements to the global evaluation process. Consequently, we have a vector with different measures from the asset's attributes. This vector is the basis to obtain the weighted quality measure of the asset.

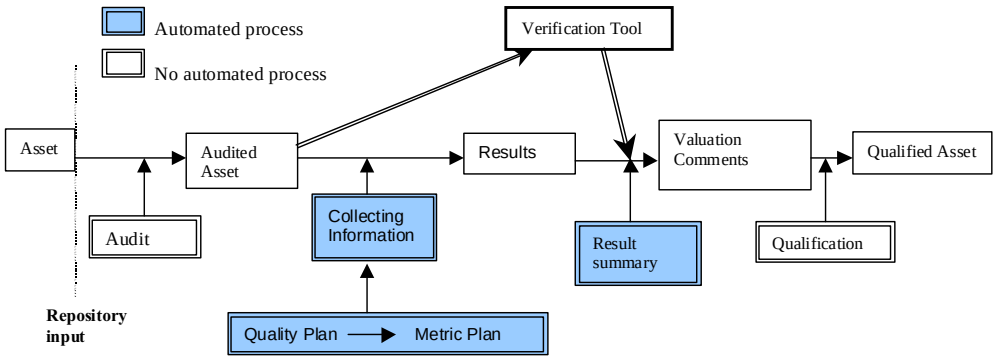

Figure 3. Qualification Process Diagram.

Finally, in the asset's retrieval aspects, faceted search methods are usually used, i.e., going with a free text (*describing the functionality of the asset*) you can give a set of values related with different sections: asset's level, quality level (*only optimum quality asset, with verification of properties, admissible quality…*) and so on.

However, the most interesting thing is the retrieval of complete mecanos, no isolated assets. A special case of the retrieval of mecanos is the generation of these. The reuse model defined by the GIRO group presents a duality compositional/generative based on mecanos [3].

The generation process, in a briefly view, consists in the selection of a set of functional or user requirements that the new mecano or *generated mecano* should fulfil or approximate in the best way. Then, taking these requirements as entry points, the generation process navigates by the asset's relationships to generate the result, following a predetermined policy where the quality information is so important. In the mecano generation area it is more important the quality of the mecano as a whole than the individual asset quality.

In this sense, we are working in the definition of an integrating schema that allows incorporation of the individual asset's properties verification to the registered[5] mecanos certification process. Other problem is derived from the automatic mecanos generation process, because the process should inform to the developer with reuse about the probable quality of a mecano, and this facility doesn't exist in the repository by now. When it exists the developer with reuse will have a comparison criterion to compare the new mecano with other existing and *validate* (*but with an apparent lower*

[4] The first audit process evaluates the syntactic correctness and completeness of the asset's documentation.
[5] In the repository.

level of usefulnes) mecanos. Perhaps, it could be more suitable a validate mecano with a 40% functional requirement fulfilment than a mecano with a 70% functional requirement fulfilment but with a doubtful quality.

5. Conclusions and future work

In this work a process model has been established to combine the UML diagrams reuse with the formal verification of their properties into a distributed context.

Also, some of the main issues to concrete this proposal have been identified: format of the diagrams, formal tests to apply, format of the verification results to be integrated into the global quality schema of the repository, and the necessary tools.

In future works new diagrams properties should be able to be verified, and more UML diagrams will be treated in the process. On the other hand, the implementation of the different tools should start to perform this schema (*a ROSE or XML to OBJ/Maude translator, an automatic properties verifier, a result generator that conforms the mecano model…*).

Another interesting future possibility will be able to be verifying a complete mecano, as an aggregation of individual interrelated assets.

Finally, another work line is the actualisation of the evaluation process when new properties for verification appear.

6. Acknowledgements

The research groups GIRO from the University of Valladolid and GIS from the University of Murcia have performed this paper in collaboration inside the MENHIR project. Special thanks are given to the other members of both groups for their help and for the fruitful and encouraging discussions about the treated themes. This work has been partially financed by the CICYT TIC97-0593-C05-05 project.

7. References

[1] Booch, G., Jacobson, I. and Rumbaugh, J. "*The Unified Modeling Language User Guide*", Addison-Wesley, 1999.

[2] García, F. J., Marqués, J. M., Laguna, M. Á. and Maudes, J. M. "*Estructuras Complejas de Reutilización: Definición de Mecano Estático*". In the proceedings of the II Jornadas de Trabajo MENHIR. Editor José A. Carsí (Valencia, 19-20 de Febrero de 1998): 135-141.

[3] García, F. J., Marqués, J. M. and Maudes, J. M. "*Mecanos as Basis of a Compositional/Generative Mixed Reuse Model*". In the proceedings of the second edition of the European Reuse Workshop (Madrid, 4-6 November, 1998). Vol.2: 17-20.

[4] García, F.J., Romay, Mª., Marqués, J.M. and Crespo, Y. "*Mecanos: Exposición de Resultados y Líneas de Trabajo Abiertas en la Reutilización Sistemática del Software*". In the proceedings of the IDEAS'99 Workshop. 1999.

[5] Meyer, B. "*The Next Software Breakthrough*". Computer. July 1997.

[6] Meyer, B, Mingins C. and Schmidt, H. "*Providing Trusted Components to the Industry*". Computer. pp. 104-105. May 1998.

[7] Clavel, M., Durán, F., Eker, S., Lincoln, P., Martí-Oliet, N., Meseguer, J. and Quesada, J. "*Maude: Specification and Programming in Rewriting Logic*", Computer Science Laboratory SRI International, January 1999.

[8] OMG. "*XML Metadata Interchange (XMI)*". Proposal to the OMG OA&DTF RFP 3: Stream-based Model Interchange Format (SMIF). October 1998.

[9] Manso E., García, F. J., Rodríguez, J. J. and Laguna, M. A. "*Modelo de Cualificación de Assets del Repositorio GIRO*". In the proceedings of the III Jornadas de Trabajo de MENHIR. B. Moros and J. Saez editors (Murcia, 13-14 November, 1998): 109-114.

[10]SER Consortium. "*Solutions for Software Evolution and Reuse***".** SER Esprit Project 9809. Jan., 1996.

[11]Fernández, J. L. and Toval, A. "*Formal Verification of UML Specifications***"** (In Spanish. Proc. III Jornadas de Ingeniería del Software, Murcia, November 1998.