OSS tools in a heterogeneous environment for embedded systems modelling: an analysis of adoptions of XMI

Anna Persson[1], Henrik Gustavsson[1], Brian Lings[1], Björn Lundell[1], Anders Mattsson[2], Ulf Ärlig[2]

[1]University of Skövde
P.O. Box 408, SE-541 28 SKÖVDE
Sweden
+46(0)500448000

{ anna.persson | henrik.gustavsson | brian.lings | bjorn.lundell }@his.se

[2]Combitech Systems AB
P.O. Box 1017, SE-551 11 JÖNKÖPING
Sweden
+46(0)36194750

{ anders.mattsson | ulf.arlig } @combitechsystems.com

ABSTRACT
The development and maintenance of UML models is an inherently distributed activity, where distribution may be geographical, temporal or both. It is therefore increasingly important to be able to interchange model information between tools – whether in a tool chain, for legacy reasons or because of the natural heterogeneity resulting from distributed development contexts. In this study we consider the current utility of XMI interchange for supporting OSS tool adoption to complement other tools in an embedded systems development context. We find that the current state of play is disappointing, and speculate that the problem lies both with the open standards and the way in which they are being supported and interpreted. There is a challenge here for the OSS community to take a lead as tool vendors gear up for XMI 2.0.

Categories and Subject Descriptors
D.2.2 [**Software Engineering**]: Design Tools and Techniques – *Computer-aided software engineering (CASE), Object-oriented design methods.*

General Terms
Design, Standardization, Management.

Keywords
Embedded Systems Modelling, XMI, Open Standards, Heterogeneous Tool Environment, Model Interchange.

Open Source Application Spaces: Fifth Workshop on Open Source Software Engineering (5-WOSSE) May 17, 2005, St Louis, MO, USA.

1. INTRODUCTION
In this paper we explore adoptions of the XMI standard interchange format [1] in UML modelling tools with a view to investigating the current practicality of supporting heterogeneous tool environments. Model interchange is important for two reasons. Firstly, it is widely acknowledged that systems outlive tools (see, for example, [2] [3]). Secondly, companies often use more than one tool in their development environments, as tools have different strength and weaknesses, perhaps at different stages in the tool chain. Here, we report on a study to identify combinations of modelling tools able to utilise XMI-based interchange of UML class diagrams in the context of embedded systems models. Our specific interest is in analysing OSS tools in this context, so we have conducted a study which incorporates all of the OSS UML modelling tools supporting XMI which are known to us.

In principle, XMI allows for the interchange of models between modelling tools in distributed and heterogeneous environments, and eases the problem of tool interoperability [1]. Most major UML modelling tools currently offer model interchange using XMI [4] [5]. XMI is an open standard and adherence to open standards has always been viewed as central to the open source movement, and a key to achieving interoperability [6, p. 83]. It is therefore expected that open source tools will display good characteristics in this respect.

The study consisted of interchanging a simple UML model between a set of OSS and proprietary UML modelling tools. The model describes part of an embedded system and is simplified from a class diagram developed by the company Combitech Systems AB. First we created the model in each of the tools, and then we exported it in XMI. All XMI documents exported were then imported into each of the tools.

2. BACKGROUND
The study explored three open source modelling tools: ArgoUML (Argo), Fujaba Developer (Fujaba) and Umbrello UML Modeller (Umbrello). These tools were selected since they support UML modelling and interchange of UML models using XMI. A systematic review of available open source modelling tools revealed no other tools with these properties. We also used four

commercial (non OSS) UML tools that support XMI and are specifically targeted at modelling embedded systems: Artisan Real-Time Studio (Artisan), Poseidon Embedded Enterprise (Poseidon, a tool which originated from the ArgoUML project), Rhapsody C++ Developer (Rhapsody) and TAU G2 (TAU). In addition, we included two common commercial (non OSS) tools for general UML modelling: Rational Rose Enterprise (Rose) and Microsoft Visio Professional (Visio). Table 1 gives an overview of all tools included in the study, describing the versions of UML and XMI supported in each tool.

Table 1. UML modelling tools explored in the study

	XMI version export	XMI version import	UML version
ArgoUML Version 0.16.1 (argouml.tigris.org)	1.0	1.0	1.3
Fujaba Developer Version 4.2.0 (www.fujaba.de)	1.2	1.2	1.3
Umbrello UML Modeller Version 1.3.2 (www.artisansw.com)	1.2	1.2	1.3
Artisan Real-Time Studio Version 5.0.22 (www.artisansw.com)	1.1	1.1	2.0
Poseidon Emb. Enterprise Version 3.0.1 (www.gentleware.com)	1.2	1.0, 1.1, 1.2	2.0
Rhapsody C++ Developer Version 5.2 (www.ilogix.com)	1.0	1.0	1.3
Rose Enterprise Version 2003.06.13 (www.rational.com)	1.0, 1.1	1.0, 1.1	1.3
TAU G2 Version 2.4 (www.telelogic.com)	No export	1.0, 1.1	2.0
Microsoft Office Visio Version Prof. 2003 (www.microsoft.com)	1.0	No import	1.3

The model interchanged between these tools is shown in figure 1. Versions of XMI earlier than 2.0 do not cater for the exchange of presentation information, so layout aspects are therefore lost at interchange. In practice, this is a significant problem where there is subsequent human interaction with the model, but of less significance for functions such as code generation.

Interchange of a model between two tools is said to be successful if all model information other than presentation information is preserved during the transfer. An interchange resulting in incomplete model information is clearly unacceptable: commercial models that often consist of several thousands model entities [7], and manual repair is infeasible.

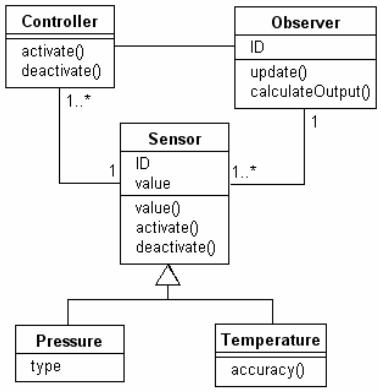


Figure 1. UML model used for interchange

3. RESULTS

Table 2 presents the results from our analysis of transfer between each pair of tools explored in the study. Non-coloured cells in the table are expected to work since the versions of XMI supported in both tools are the same. Grey cells (*italic*) are not expected to work since the XMI versions used in the two tools differ.

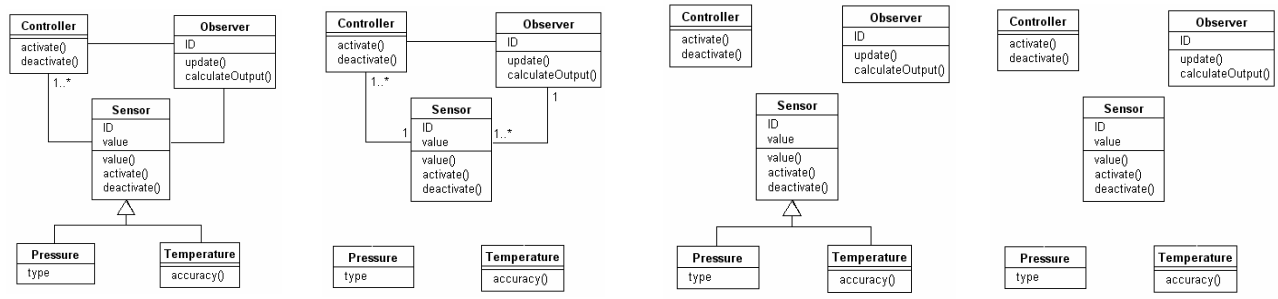
For combinations of tools resulting in incomplete model interchange figure 2 shows what remains of the model after transfer. Four types of unsuccessful transfer were identified: three cardinalities lost (a); all inheritance relations lost (b); all associations lost (c); and all relations lost (d). For other combinations of tools resulting in unsuccessful model interchange (i.e. for cells only containing "N" in figure 2) no model information was transferred.

Our results show unsuccessful interchange for the majority of tool combinations. We were unable to find any combination of tools that supported a two-way interchange; not even indirectly via a third tool. However, we found some combinations of tools where successful one-way interchange is possible, even though most successful interchanges in table 2 refer to interchange within the same tool (which we analysed as a basic functionality test of exporting and importing XMI files in a tool).

For interchange between tools using the same XMI version we found both successful (e.g. when exporting from Argo and importing to TAU) and unsuccessful transfer (e.g. when exporting from Fujaba and importing to Umbrello). For some of the successful combinations, the exporting and importing tools support the same version of UML (e.g. when exporting from Rhapsody and importing to Argo), whereas other successful combinations support different versions of UML (e.g. when exporting from Fujaba (1.3) and importing to TAU (2.0)). Further, when interchanging between tools using different XMI versions we found both successful (e.g. when exporting from Fujaba and importing to TAU) and unsuccessful transfer (e.g. when exporting from Artisan and importing to Umbrello).

Table 2: All combinations of successful interchange between tools (Y=Yes, N=No).

Import → / Export ↓	Argo	Fujaba	Umbrello	Artisan	Poseidon	Rhapsody	Rose	TAU	Visio
Argo	**Y**	*N*	*N*	*N*	**Y**	N,c	N,a	**Y**	---
Fujaba	*N*	**Y**	N	*N,d*[1]	N	*N*	*N*	**Y**	---
Umbrello	*N*	N	**Y**	*N*	N	*N*	*N*	*N*	---
Artisan	*N*	*N*	*N,d*	N,b	N	*N*	*N,c*	N,b	---
Poseidon	*N*	N	N,d	*N,b*[1]	**Y**	*N*	*N,c*	*Y*	---
Rhapsody	**Y**	*N*	*N*	*N*	N	**Y**	N	**Y**	---
Rose 1.0/1.1	**Y** *N*	*N* *N*	*N* *N*	*N* N,b[1]	**Y** N,c	N,c *N*	**Y**	**Y** **Y**	---
TAU	---	---	---	---	---	---	---	---	---
Visio	**Y**	*N*	*N*	*N*	N	N	N	**Y**	---



a: Three cardinalities lost **b**: Inheritance relations lost **c**: Associations lost **d**: All relations lost

Figure 2 a, b, c, d: Model information transferred.

4. ANALYSIS

The results presented in section 3 show that XMI-based model interchange between UML modelling tools is weakly supported in practice. A contributing factor is that tools supporting different XMI versions cannot interchange their XMI documents. This has earlier been reported in the literature as causing incompatibility between tools [8] [9]. All three open source tools analyzed in this study support only one version each of XMI. We view this as a weakness that will limit the inclusion of these tools in heterogeneous tool environments.

Another aspect that further complicates model interchange is that different versions of UML are supported by different tools. A tool supporting an earlier version of UML may have problems importing XMI documents exported from a tool supporting a later version of UML. The different versions of both UML and XMI lead to a large number of possible combinations, some of which may be incompatible.

Compatibility between XMI and UML versions are, however, no guarantee for successful interchange between tools, as shown by the results in section 3. We note that XMI export has been implemented in a variety of different ways amongst the tools analyzed. For example, a comparison between Argo and Visio (two tools supporting the same versions of UML and XMI) shows that Visio exports an XMI document with more than seven times as many rows as the XMI document exported from Argo (4967 rows vs. 651 rows).

For each tool supporting XMI 1.0 we analysed the exported document using an independent XML validation tool (www.altova.com) utilising OMG's normative DTD. The documents from Argo, Rhapsody and Rose were found to be valid, that from Visio was not. For later versions of XMI there is some uncertainty concerning which DTD should be used and we were unable to successfully use the DTD presented in the XMI specification.

5. SUMMARY

Adoption of new tools into existing systems development contexts will depend heavily on their ability to interchange models. For a company wishing to base its tool strategy on open standards, our findings are rather discouraging. None of the tools analysed has adopted the latest version of XMI (2.0), and we were unable to find any combination of tools which successfully interchanged model information between them. Our findings suggest that a strategy for a company working in the context of embedded systems design cannot currently rely on standardised model interchange between different UML modelling tools.

[1] Function "value()" in class Sensor was renamed to "Operation1()".

Although OSS tools offer support for XMI-based model interchange equal to that in commercial tools, better could be expected. Commercial tools offer proprietary bridges to other tools, particularly market leaders, and may even make efforts to improve XMI interchange by catering for product-specific interpretations of XMI. However, the OSS community can be expected to offer high conformance with any open standard, and not to resort to tool-specific bridging software. Further, it could be argued that a goal for OSS tools should be to offer reliable import and export of documents conforming to any of the XMI versions, in this way offering both openness and an important role in the construction of interchange adapters – especially useful for legacy situations. As a special case, one hopes that OSS tools will lead the way in conformance with XMI 2.0. With the advent of UML 2 and XMI 2.0, there is a real possibility of standard interchange both horizontally and vertically within the tool chain.

As long as successful model interchange can be provided between tools supporting XMI 1.x the fact that layout aspects are lost during interchange does not necessarily hinder their use in a tool chain. For example, in a scenario in which developers design their models in one tool and use a different tool for code generation, the loss of layout information may not be critical. However, in such one-way exchange the company must consider increased complexity in model management, and there may be an increased risk associated with such a solution from a maintenance perspective (together with practical problems related to upgrade of a tool in the chain without breaking the chain). Of course, as different designers have different preferences, a tool chain may even involve yet further tools, for example for use in debugging activities.

The issue of conformance is so central to effective interchange that much greater support needs to be given to conformance checking, to aid manufacturers aiming for tool openness. For example, it needs to be clear what is the official DTD for each version of XMI and UML and what is guaranteed by any claim to conformance for each version. It would also be useful for the community to develop benchmark interchange models – rather richer than the class model used in this study.

We do not put this study forward as a definitive review of the tools involved. For example, later versions of (or plug-ins for) some tools are now available in which improved XMI support is evident; and the use of industrial strength diagrams will lead to more realistic scenarios.

However, we have contacted all the manufacturers involved in the study and received many detailed responses concerning our findings. It is clear that the results do not come as a surprise, and many reasons for this have been put forward. These range from ambiguities in the standard to problems in supporting the parsing of the many legal formats still allowed by a DTD. Many manufacturers are now working towards XMI 2.0, and placing increasing emphasis on compliance – in at least one case thus reducing conformity with one of the market leaders. It is our hope that the OSS community will take a lead in this important next stage.

6. ACKNOWLEDGMENTS

This research has been financially supported by the European Commission via FP6 Co-ordinated Action Project 004337 in priority IST-2002-2.3.2.3 ‘Calibre’ (http://www.calibre.ie).

7. REFERENCES

[1] OMG-XML Metadata Interchange (XMI) Specification, version 1.0-2.0 http://www.omg.org/technology/documents/modeling_spec_catalog.htm#XMI.

[2] Lundell, B. and Lings, B. Changing perceptions of CASE-technology, *Journal of Systems and Software*, 72, 2 (2004), 271-280.

[3] Lundell, B. and Lings, B. Method in Action and Method in Tool: a Stakeholder Perspective, *Journal of Information Technology*, 19, 3 (2004), 215-223.

[4] Jeckle, M. OMG’s XML Metadata Interchange Format XMI. In *Proceeding of XML Interchange Formats for Business Process Management (XML4BPM 2004): 1st Workshop of German Informatics Society e.V. (GI)* (in conjunction with the 7th GI Conference “Modellierung 2004”), Marburg, Germany, 25 March 2004.

[5] Stevens, P. Small-scale XMI programming: a revolution in UML tool use? *Automated Software Engineering*, 10, 1 (2003), 7-21.

[6] Fuggetta, A. Open Source Software: An Evaluation, *Journal of Systems and Software*, 66, 1 (2003), 77-90.

[7] Berenbach, B. The Evaluation of Large, Complex UML Analysis and Design Models. In *Proceedings of 26th International Conference on Software Engineering (ICSE’04)*, IEEE Computer Society, Los Alamitos, 2004, 232-241.

[8] Damm, C.E., Hansen, K.M., Thomsen, M. and Tyrsted, M. Tool Integration: Experiences and Issues in Using XMI and Component Technology. In *Proceedings 33rd International Conference on Technology of Object-Oriented Languages and Systems TOOLS 33*, IEEE Computer Society, Los Alamitos, 2000, 94-107.

[9] Jiang, J. and Systä, T. (2003) Exploring Differences in Exchange Formats – Tool Support and Case Studies. In *Proceedings of Seventh European Conference on Software Maintenance and Reengineering (CSMR’03)*, Benevento, Italy, March 26-28, 389-398.