Making UML models interoperable with UXF

Junichi Suzuki and Yoshikazu Yamamoto

Department of Computer Science,
Graduate School of Science and Technology,
Keio University
Yokohama City, 223-8522, Japan.
+81-45-563-3925 (Phone and FAX)
{suzuki, yama}@yy.cs.keio.ac.jp,
http://www.yy.cs.keio.ac.jp/~suzuki/project/uxf

Abstract. Unified Modeling Language (UML) has been widely accepted in the software engineering area, because it provides most of the concepts and notations that are essential for documenting object-oriented models. However, UML does not have an explicit format to describe and interchange its model information intentionally. This paper addresses the UML model interchange and presents our efforts to make UML highly interoperable. We developed an interchange format called UXF (UML eXchange Format) based on XML (Extensible Markup Language). UXF is a simple and well-structured format to encode UML models. It leverages the tool interoperability, team development and reuse of design models by interchanging the model information with the the XML standard. Also, we propose an open distribution platform for UML models, which provides multiple levels of interoperability of UML models. Our work shows an important step in the evolution for the interoperable UML.

1 Introduction

Unified Modeling Language (UML) [1–8] has been widely accepted as an object oriented software analysis/design methodology in the software engineering community. It provides most of the concepts and notations that are essential for documenting object oriented models. While UML is the union of the previously leading object modeling methodologies; Booch [9], OMT [10] and OOSE [11], it includes additional constructs that these methods did not address, such as Object Constraint Language (OCL) [6] and Object Analysis & Design CORBAfacility Interface Definition [8]. It is the state of the art convergence of practices in the academic and industrial community. Also, as a publicly available standard, UML is now in the process of standardization and revision at Object Management Group (OMG) [12].

UML provides a series of diagrams with the fine level of abstraction to specify object models for a given problem. Complex systems can be modeled through a small set of nearly independent diagrams. UML defines two aspects for constructs in every diagram:

- Semantics:
  The UML metamodel defines the abstract syntax and semantics of object modeling concepts.
- Notations:
  UML defines graphical notations for the visual representation of its model elements.

While UML defines coherent model elements and their interchangeable semantics, it does not intentionally provide the explicit format to exchange the model information. The ability of model interchange is quite important because it is likely a development team resides in separate places on a network environment, and because current UML models are not interoperable between development tools due to the lack of an application-neutral exchange format [13].

This paper addresses the standard-based UML model interchange and presents our efforts to make UML models interoperable. We have developed an interchange format called UXF (UML eXchange Format) [14], which is based on XML (eXtensible Markup Language). We consider the use of XML as a mechanism for encoding and exchanging the structured data defined with UML.

The remainder of this paper is organized as follows. Section 2 discusses some candidate formats and their pros and cons. Then, we describe rationale and merits to employ XML. Section 3 outlines comparisons with related work. Section 4 defines the scope and syntax of UXF and presents some examples of processing UXF formatted data. Section 5 presents our applications using UXF. We conclude with a note on the current status of project and future work in Section 6 and 7.

2 UML model interchange

The most important factor in interchanging UML models is the semantics within models should be described explicitly and transferred precisely. This section describes why we chose XML from some candidates, and presents the characteristics of UXF.

2.1 Candidate formats

There are some candidate formats to encode and interchange UML models. The following sections discuss their pros and cons.

Proprietary format The first candidate is a proprietary format. It is a straightforward strategy, and allows development tools to use a certain optimized syntax to encode model information. However, it suffers from non-interoperability: the model information cannot be reused between different tools. Though some tools (e.g. CASE tools) provides export/import capabilities that translate a proprietary format into another one, these are not substantial solution for the UML model interchange.

HTML (HyperText Markup Language) The second candidate is HTML. It is an easy to learn format, and has been widely accepted in the Web and documentation community. HTML, however, cannot describe arbitrary or complex data structure because it provides fixed set of tags. An example of HTML documentation tools is `javadoc` included in Java Development Kit (JDK), which is a translator from the comments in Java source code into specification documents written in HTML. While such a tool is valuable and helpful for everyday development work, some important model information is unfortunately thrown away in the process of producing HTML documents, due to its fixed tag set. In other words, HTML documents generated by documentation tools cannot describe semantics of model information precisely and also cannot be reused in other applications.

XML (eXtensible Markup Language) XML is an emerging data format in the Web community, which is standardized by the World Wide Web Consortium (W3C) [14]. While HTML is defined by SGML (Standard Generalized Markup Language: ISO 8879), XML is a sophisticated subset of SGML, and designed to describe arbitrary structures of documents beyond HTML. One of the goals of XML is to be suitable for use on the Web; thus to provide a generic mechanism for the delivery of information over the Internet. XML has the following characteristics:

– application neutrality (vender independence)
– user extensibility
– ability to represent arbitrary and complex information
– validation scheme of data structure
– human readability

As its name implies, extensibility is a key feature of XML; users or applications are free to declare and use their own tags and attributes. Therefore, XML ensures that both the logical structure and content of semantically rich information can be retained. It emphasizes the description of information structure and content as distinct from its presentation. The data structure and its syntax are defined in a DTD (Document Type Definition) specification, which specifies a set of tags and their constraints. Every XML documents can validate its content structure by comparing with its DTD. XML is also the text-based format. This means the editing of XML documents are easy and existing text manipulation tools can be used to process them.

In contrast to data structure, the presentation issue is addressed by XSL (XML Style Language) [15], which is also a W3C's standard to describe stylesheets for XML documents. XSL is based on DSSSL (Document Style Semantics and Specification Language ISO/IEC 10179) and complement CSS (Cascading Style Sheet) [16], which is a style definition language for HTML. In addition, XPointer [17] and XLink [18] are also in the process of standardization at W3C, which are specifications to define anchors and links within or across XML documents.

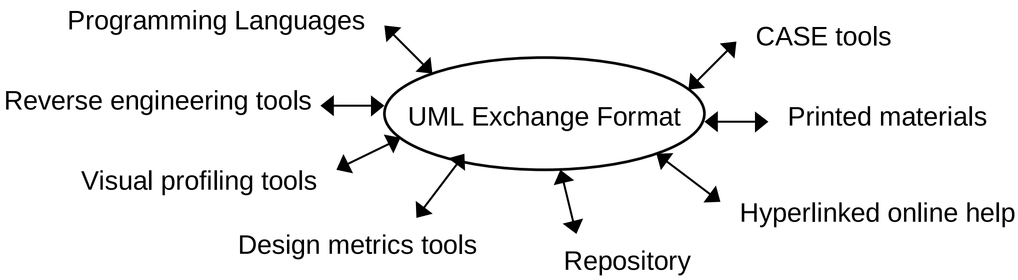

Fig. 1. UXF allows the seamless interchange of UML model information between development tools

2.2 UML eXchange Format (UXF)

As such, XML has great potential as an interchange format for UML. We have developed a XML-based format called UXF (UML eXchange Format). UXF facilitates:

- Interoperability between development tools:
  Software models are dynamically changed in the analysis/design, revision and maintenance phases, and the software tools used by a development team employ their own proprietary formats to describe the model information. UXF allows UML models to be interoperable between development tools throughout the lifecycle of software development. Once encoded with a certain format, the model information can be reusable for a wide range of different development tools with different strengths (Fig. 1). This seamless interoperability increases our productivity of UML modeling.
- Intercommunications between software developers:
  The Internet is a promising infrastructure to distribute and share software model information, because it is effective and economical for making information available to the separated group of individuals. Within the Internet/Intranet environment, especially the Web environment, we can represent and communicate software modeling insights and understandings with each other. For example, We may write down model information into electronic mails, or use a distributed communication system to transfer UXF descriptions. UXF simplifies the circulation of UML models between software developers.
- Natural extension from the existing Web environment:
  UXF is a natural and transparent extension from the existing Web environment. Thus, it allows to edit, publish, access and exchange the UXF description as easily as is currently possible with HTML. In addition, most of the existing Web applications can be used for handling UXF with relatively minor modifications.

To author and view UML models encoded with UXF, existing markup languages could be converted to UXF, and most development tools such as CASE tools, documentation tools, visual profiling tools and document repositories, can be modified so that they recognize UXF. In the current situation where many XML-aware applications exist, it is relatively easy to extend these tools. Also, UXF descriptions can be handled by every Web application that manipulates HTML as well as Web browsers/servers in the near future.

UXF also ensures a variety of possibilities of its output representations by applying different stylesheets to a UXF documnents. Output formats include RTF (Rich Text Format), HTML, LaTeX, PDF (Portable Document Format). Moreover, UXF data can embed hyperlinks using the linking mechanisms of XPointer and XLink. This allows us to link UML model elements. As such, developers can use technical materials as printed, electronic or interactive documents (Fig. 1).

3 Related work

A well-known and mature format for exchanging the software modeling information is CDIF (CASE Data Interchange Format) [19]. CDIF is a generic mechanism and format to interchange the software models between CASE tools, and a family of standards defined by the Electronic Industries Association (EIA) and International Standard Organization (ISO). CDIF defines a meta-metamodel, a tool interchange format, and a series of subject areas:

- CDIF Framework for Modeling and Extensibility
- CDIF Transfer Format
- General Rules for Syntaxes and Encodings
- SYNTAX.1
- ENCODING.1
- CDIF Integrated Metamodel
- Foundation Subject Area
- Common Subject Area
- Data Modeling Subject Area
- Data Flow Model Subject Area
- Data Definition Subject Area
- State/Event Model Subject Area
- Presentation Location and Connectivity Subject Area

CDIF separates the semantics and syntax from the encoding, and thus provides flexibility in the representation and transfer mechanism. SYNTAX.1 and ENCODING.1 defines the means that allows for a tool-independent exchange of models. CDIF has provided the mapping to UML [20], by using the Foundation Subject Area and CDIF Transfer Format, and by defining the UML subject area that provides the definitions of metamodel entities and their relationships in UML. The UML Subject Area is dependent on the CDIF Foundation Subject Area.

UXF is a UML-specific exchange format and an alternative vehicle to transfer UML models. Since it is a straightforward extension from and transparent to the Web distributed environment, it can be easy-to-learn for the huge amount of people that are familiar with HTML or SGML. We believe UXF is much easy and practical approach for interchanging UML models over the Internet. We are also investigating the possibility to integrate UXF with the CDIF effort (see Section 6).

As described in Section 1, UML is now in the process of revision. As for model interchange, OMG issued a RFP (Request For Proposal) for SMIF (Stream-based Model Interchange Format) specification [21]. Responses for SMIF include CDIF based, STEP based and XML based proposals. At present, UXF is not compliant to SMIF intentionally for the simplicity of the format. SMIF is just proposed and has not been frozen, at the time of this writing. Once SMIF is frozen or more mature, we will develop a translator between UXF and SMIF. UXF is carefully designed to be [13, 22, 23]:

- Simple: UXF is compact by including only UML's semantics, while the scope of SMIF includes other specification (e.g. Meta Object Facility).
- Intuitive: UXF is easy-to-learn and readable.
- Lightweight: The intention of UXF does not include only an interchange format, but also more broad range of interoperability for UML models (see Section 6). UXF serves as a lightweight means for such usage.

These characteristics are also strength for other description formats for UML [24, 25].

4 UXF design principle

In terms of interchanging model information between development tools, there can be two types of information that should be exchanged [20]:

- Model-related information
- View-related information

While model-related information is a series of building blocks to represent a given problem domain (e.g. classes, attributes and associations), view-related information is composed of the way in which the model elements are rendered (e.g. the shapes and position of graphical objects). This paper concentrates on exchanging model-related information. The interchange of view-related information is out of the scope of our work. However, it is easy to obtain the view-related information by generating a data description for a certain rendering application, or applying XSL stylesheets to UXF.

4.1 UXF DTDs

The UXF specification actually consists of a series of XML DTDs. It provides the mapping of UML model elements into XML tags. UXF captures the model elements in the UML metamodel and defines each as a tag (or document element) straightforwardly. The attributes of each UML element are mapped into attributes of the corresponding UXF tag.

We have specified UXF DTDs for essential diagrams for the analysis and design: Class, Collaboration and Statechart diagrams. Table 1 depicts the mapping of UML model elements and UXF tags. Current UXF supports most elements in the Core, Collaboration, State Machines package and some elements in other packages in UML version 1.1. Using UXF, most essential concepts and constructs in UML can be mapped to the stream-based description seamlessly. Complete DTDs, sample markup examples and other materials can be found at [26]. Note that constructs described with UXF are not shared between different diagrams. Section 6 describes this issue.

UML Package	**UML model element**	**UXF tag**
Core	Association	<Association>
	AssociationEnd	<AssocRole><PeerAssocRole>
	Attribute	<Attribute>
	Class	<Class>
	Dependency	<Dependency>
	Generalization	<Generalization>
	Interface	<Interface>
	Operation	<Operation>
	Parameter	<Parameter>
Auxiliary Elements	Refinement	<Refinement>
Extension	TaggedValue	<TaggedValue >
Common Behavior	Exception	<Exception>
	Action	<Action>
	ActionSequence	<ActionSequence>
	Instance	<Instance>
Model Management	Model	<Model>
	Package	<Package>
Collaborations	Collaboration	<Collaboration>
	Interaction	<Interaction>
	Message	<Message>
StateMachines	CompositeState	<CompositeState>
	Event	<Event>
	Guard	<Guard>
	State	<State>
	Transition	<Transition>
	PseudoState	<PseudoState>

Table 1. Comparison between UML model elements and UXF tags

4.2 Processing UXF documents

This section outlines how a UXF documents might be processed. In every phase, we can reuse various existing XML or SGML tools.

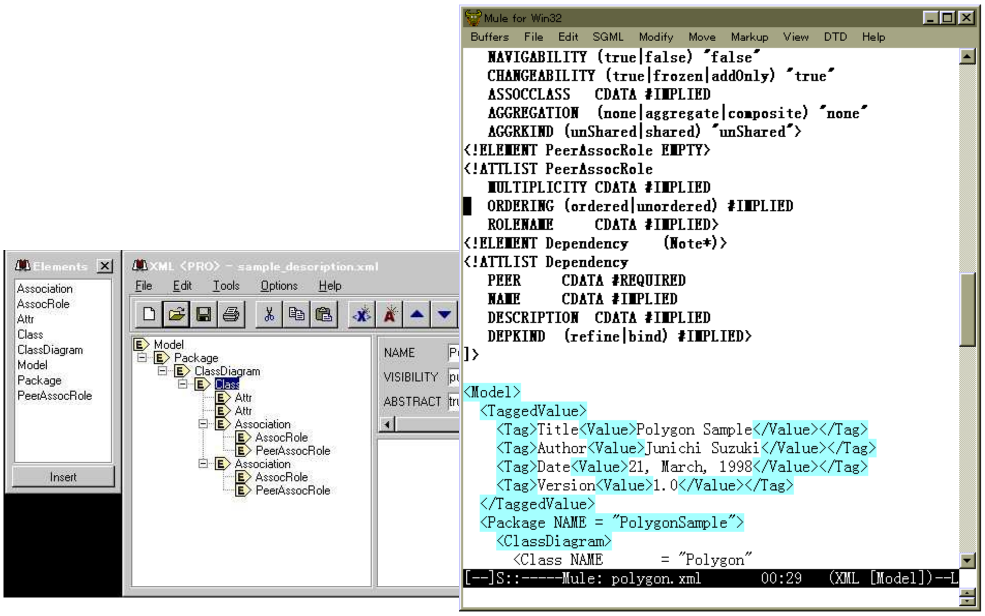

Fig. 2. Sample screenshots of XML editors editing UXF descriptions (XML Pro from Vervet logic and a XML major mode for Emacs named psgml)

Authoring UXF description can be created with any text editor because it is a text-based format. In practice, it is expected to use an editing tool that helps users' input. Figure 2 shows sample screenshots of commercial and freely available XML editors editing UXF description.

Conversion Data conversion makes the authoring work simple and productive. UXF description can be converted from/to other data (e.g. legacy documents, program source code, documentation format or data representation in a development tools). UXF allows such conversion programs to be written easily. Examples are described in Section 5.

Parsing Parsing is the process to analyze and validate the syntax of UXF documents. XML allows for two kinds of descriptions; valid and well-formed. Validity requires that a document refers a proper DTD and obeys its constraints. Well-formedness is a less strict criteria and requires that a document just obeys the syntax of XML. UXF requires a validating parser in authoring UXF descriptions, and a non-validating parser in browsing or delivering the document. We can use any XML parser from huge amount of existing parsers.

Distribution UXF is designed to distribute UML models precisely over the network environment. It can be used in existing document distribution systems to

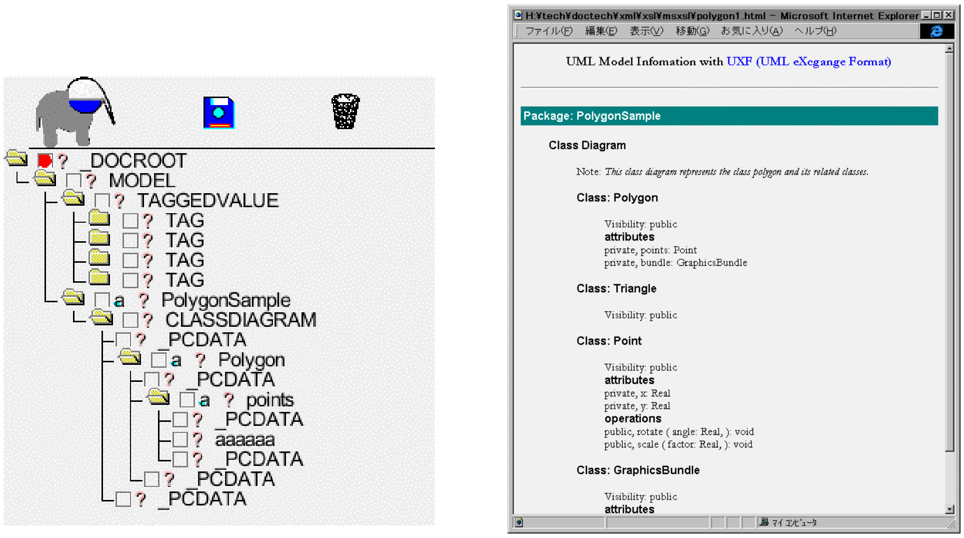

Fig. 3. Sample screenshots of XML browsers rendering UXF descriptions (a XML viewer named Jumbo and Microsoft Internet Explorer

share and manage UML model information. Also, it can be used on the existing Web environment so that a Web browser downloads UXF description and displays them using a stylesheet or Java applets. We have developed a distributed management system for UML models (see Section 5).

Rendering and Browsing Rendering and browsing involves the delivery of stylesheets or any specialized software for display such as Java applets (see also Section 5). Figure 3 shows a Web browser that displays a UXF document using a XSL stylesheet, and a hierarchical XML browser.

5 Applications

This section presents our applications using UXF. These applications show the potential of UXF and provide standard-based ways to share UML models between various tools or over the distributed network environment.

5.1 Source code documentation tools

In general, source code documentation tool is generally a tool that imports the program source code and generates documents, along with any specialized format. We have developed a documentation tool that parses source code written in Java and generates UXF formatted documents. This tool uses Java Development Kit (JDK) by creating a class `UxfDocumentationGenerator` that extends

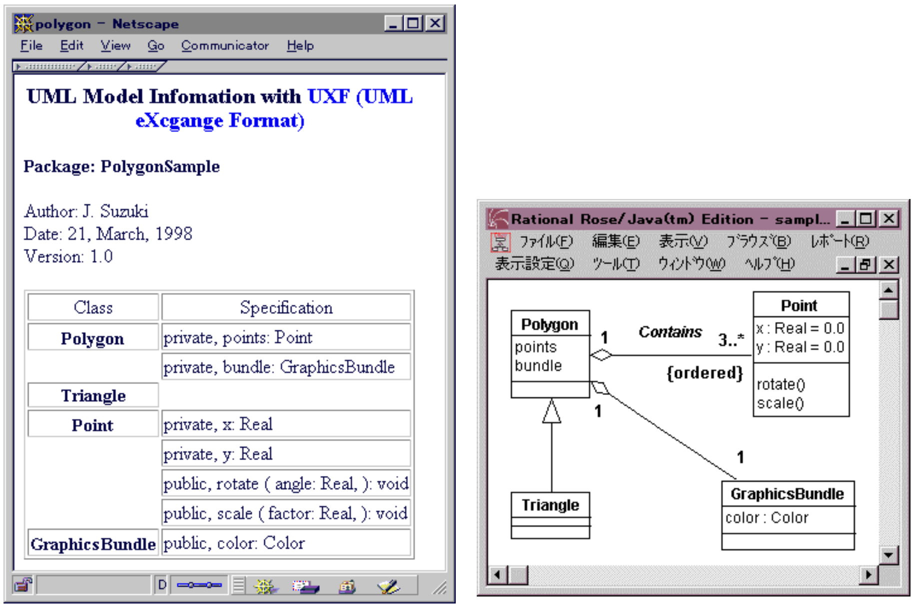

Fig. 4. Sample screenshots of UXF applications (Netscape Communicator and Rational Rose)

`DocumentationGenerator` included in JDK [13]. It translates the declaration in a Java program into the corresponding UXF representation based on the mapping in Table 1. This tool allows the model information obtained from source code to be reusable for other applications including CASE tools and repositories.

5.2 Translator between UXF and CASE tools

This tool is a translator that converts a UXF description into a proprietary format of a CASE tool, vice versa. This sort of tool is highly required, because CASE tools are frequently used in many development projects. Our tool generates the importable files (*.mdl files) of Rational Rose [22]. The left of Figure 4 shows a screenshot that Rational Rose displays a class diagram converted from a UXF description (The graphical position of classes, associations and labels are moved manually for the readability.).

5.3 Distributed model management system

The last application is a distributed model management system that shares and manages UML design information within a networked environment. Our system leverages the team development that allows developers to continue their work concurrently at the physically separated places. We have developed this system on top of the existing Web environment and a Java-based ORB (Object Request

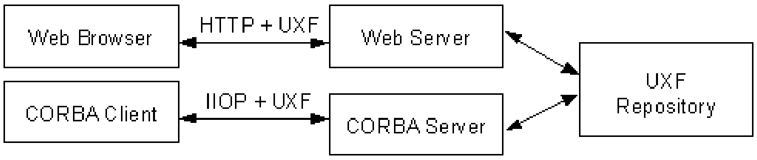

Fig. 5. Deployment architecture of our distributed model management system

Broker) compliant to CORBA (Common Object Request Broker Architecture), which is a standard for the distributed object middleware [27]. It is based on the three-tier deployment architecture, and provides two kinds of accesses to UXF documents; via HTTP and IIOP (Internet Inter-ORB Protocol), which is a TCP/IP based standard protocol of CORBA (Fig. 5). The communications via IIOP is achieved through the CORBA standard IDL (Interface Definition Language) interfaces (Fig. 6).

The HTTP access aims to allow client applications including Web browsers to refer the UXF documents that are stored in Web servers or any backend databases. Figure 3 and Figure 4 show Web browsers that display a UXF description together with different XSL stylesheets. As such, different presentations suited to the specific purpose can be displayed, if different stylesheets are prepared [28]. We have also developed a Java applet to display a graphical representation for UXF [28].

The IIOP access aims to allow developers at separated places to consistently register, refer, process and change UXF descriptions. As depicted in Figure 6 our system transfers UXF descriptions through interfaces provided by Document Object Model (DOM), a standard of the World Wide Web Consortium (W3C) [29]. DOM defines a set of interfaces to manipulate the content, structure and style of XML/HTML documents. We implemented DOM APIs on top of CORBA [23]. By combining promising standards, we achieved an open system allowing UML models to be interoperable in a distributed environment. A server application parses UXF documents at the system's start-up time or on the fly, and creates their in-memory structures; tree structures of parsed UXF elements. Client applications include simple command-line tools, GUI profiling tools and development environments [23].

6 Current ongoing projects and future work

UXF currently supports class, collaboration and statechart diagrams. We are developing DTDs for all the UML diagrams. Also, we are investigating a translator from/to the CDIF XML-based Transfer Format [30] and the possibility of the integration with it. Also, we plan to use XPointer and XLink to connect logically same model elements in different diagrams, because UXF descriptions are not shared across diagrams, as described in Section 4.1.

As for UXF-aware tools, UXF converters from/to C++, Smalltalk, Python and CORBA IDL are currently developed. The model information implemented

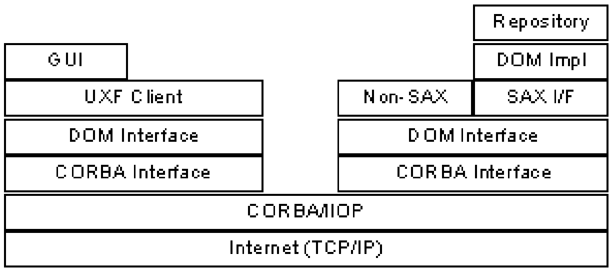

Fig. 6. Layered architecture based on DOM and CORBA

in different programming languages can be fully interoperable between different development tools by using multiple source code generation tools. A diagram editing/drawing tools are also planned.

As for distributed model management system, we are investigating to use an object-oriented database for a persistent storage of UXF descriptions. It enhances the current transient CORBA severs to be a persistent, which can maintain the tree structures of parsed UXF elements even after the shutdown of a server. Another enhancement is to provide a capability of the revision control for UXF using two XML tags; <![INCLUDE[...]]> and <![IGNORE[...]]>.

We are working for some further projects that enhance the interoperability of UML models. Our goal is to provide multiple levels of interoperability for UML. Three levels of interoperability [23] are achieved at present:

- UXF: UXF allows UML models to be interoperable between UML compliant tools.
- DOM: DOM allows UXF descriptions (virtually XML documents) to be interoperable between XML compliant tools through the uniform interface.
- CORBA: CORBA provides the standard interfaces to allow DOM compliant tools to interact with each other on the network environment, thereby UXF descriptions can be transferred between distributed DOM compliant tools.

We will use UXF as a lightweight interchange format for a testbed to enhance UML with emerging standards and technologies.

7 Conclusion

This paper addressed how UML models can be interoperable and proposed a solution that provides an standard-based format called UXF. We also proposed an open environment for highly interoperable UML models by combining some emerging standards: XML, DOM and CORBA. With UXF, UML models can be distributed universally. Our work shows how UML compliant tools can be used in the near future, and provides a blue print indicating the evolution for the interoperable UML. Information on our project can be obtained at [26].

References

1. Rational Software et.al. *UML Proposal Summary*. OMG document number: ad/97-08-02, 1997.
2. Rational Software et.al. *UML Summary*. OMG document number: ad/97-08-03, 1997.
3. Rational Software et.al. *UML Semantics*. OMG document number: ad/97-08-04, 1997.
4. Rational Software et.al. *UML Notation Guide*. OMG document number: ad/97-08-05, 1997.
5. Rational Software et.al. *UML Extension for Objectory Process for Software Engineering*. OMG document number: ad/97-08-06, 1997.
6. Rational Software et.al. *Object Constraint Language Specification*. OMG document number: ad/97-08-08, 1997.
7. Rational Software et.al. *UML Extension for Business Modeling*. OMG document number: ad/97-08-07, 1997.
8. Rational Software et.al. *OA&D CORBAfacility*. OMG document number: ad/97-08-09, 1997.
9. G. Booch. *Object-Oriented Analysis and Design 2nd edition*. The Benjamin/Cummings Publishing, 1994.
10. J. Rumbaugh et.al. *Object-Oriented Modeling and Design*. Prentice Hall, 1991.
11. I. Jacobson. *Object-Oriented Software Engineering: A Use Case Driven Approach*. Addison-Wesley, 1995.
12. UML Revision Task Force in Object Management Group at http://uml.systemhouse.mci.com/.
13. J. Suzuki and Y. Yamamoto. Making UML models exchangeable over the internet with XML. In *Proceedings of UML '98*, pages 65–74, Mulhouse, France, June 1998.
14. T. Bray J. Paoli and C. M. Sperberg-McQueen (eds.). *Extensible Markup Language (XML) 1.0*. W3C Recommendation 10-February-1998, http://www.w3.org/TR/1998/REC-xml-19980210, 1998.
15. J. Clark and S. Deach (eds.). *Extensible Stylesheet Language (XSL)*. W3C Working Draft 18-August-1998, http://www.w3.org/TR/WD-xsl, 1998.
16. B. Bos H. W. Lie C. Lilley and I. Jacobs (eds.). *Cascading Style Sheets, level 2: CSS2 Specification*. W3C Recommendation 12-May-1998, http://www.w3.org/TR/REC-CSS2/, 1998.
17. E. Maler and S. DeRose (eds.). *XML Pointer Language (XPointer)*. W3C Working Draft 03-March-1998, http://www.w3.org/TR/1998/WD-xptr-19980303, 1998.
18. E. Maler and S. DeRose (eds.). *XML Linking Language (XLink)*. W3C Working Draft 03-March-1998, http://www.w3.org/TR/1998/WD-xlink-19980303, 1998.
19. A series of CDIF specifications are available at http://www.cdif.org/.
20. Rational Software. *UML-Compliant Interchange Format*. OMG document number: ad/97-01-13, 1997.
21. Object Management Group. *Stream based Model Interchange Format (SMIF) specification RFP*. OMG document number ad/97-12-03, http://www.omg.org/library/schedule/Stream-based_Model_Interchange.htm, 1998.
22. J. Suzuki and Y. Yamamoto. Managing the software design documents with XML. In *Proceedings of the 16th Annual International Conference of Computer Documentation (ACM SIGDOC '98)*, pages 127–136, Quebec City, Canada, September 1998.

23. J. Suzuki and Y. Yamamoto. Toward the interoperable software design models: quartet of UML, XML, DOM and CORBA. In *Proceedings of the 4th IEEE International Software Engineering Standards Symposium (ISESS '99)*, to be appeared, May 1999.
24. UML Xchange. at http://www.cam.org/ nrivard/uml/umlxchng.html.
25. UML to Text. at http://www.ccs.neu.edu/home/nickman/com1205/uml-text.html.
26. UXF project Web site. at http://www.yy.cs.keio.ac.jp/ ~suzuki/project/uxf.
27. Object Management Group. *Common Object Request Broker Architecture version 2.2*. available at http://www.omg.org/, 1998.
28. J. Suzuki and Y. Yamamoto. Document brokering with agents: Persona approach. In *Proceedings of Workshop on Interactive System and Software (WISS) '98*, to be appeared, December 1998.
29. V. Apparao et al (eds.). *Document Object Model (DOM) Level 1 Specification version 1.0*. W3C proposed recommendation, 18 August 1998, 1998.
30. The CDIF XML-based Transfer Format. at http://www.cdif.org/overview/xmlsyntax.html.