UN/CEFACT/TMWG OO-edi compatibility with XML/EDI.

The XML/EDI Group,

July, 1998

Authors / Contributors:

David Webber
Klaus-Dieter Naujok

Version: 0.92

TABLE OF CONTENTS

Introduction

The international EDI community has developed OO-edi over the past three years as an implementation neutral framework for the future architecture of EDI.

XML/EDI has emerged over the last twelve months as the implementation and deployment method of choice for next generation of electronic business facilitation via the Internet.

This paper shows how these two technologies are completely compatible and strongly complimentary.

XML/EDI emerges as the implementation method of choice for FVS (Functional Service View) of Open-edi, and Open-edi BOV (Business Operational View) provides the means to underpin the physical implementation methods of XML/EDI with design, process and logic verification to ensure robust and efficient systems development and architectures.

Two representation methods provide the link between the two:

- Universal Model Language (UML)[1]
- extensible Markup Language (XML).

Figure 1 represents this ability to have a two way interchange based on the transfer of representations.

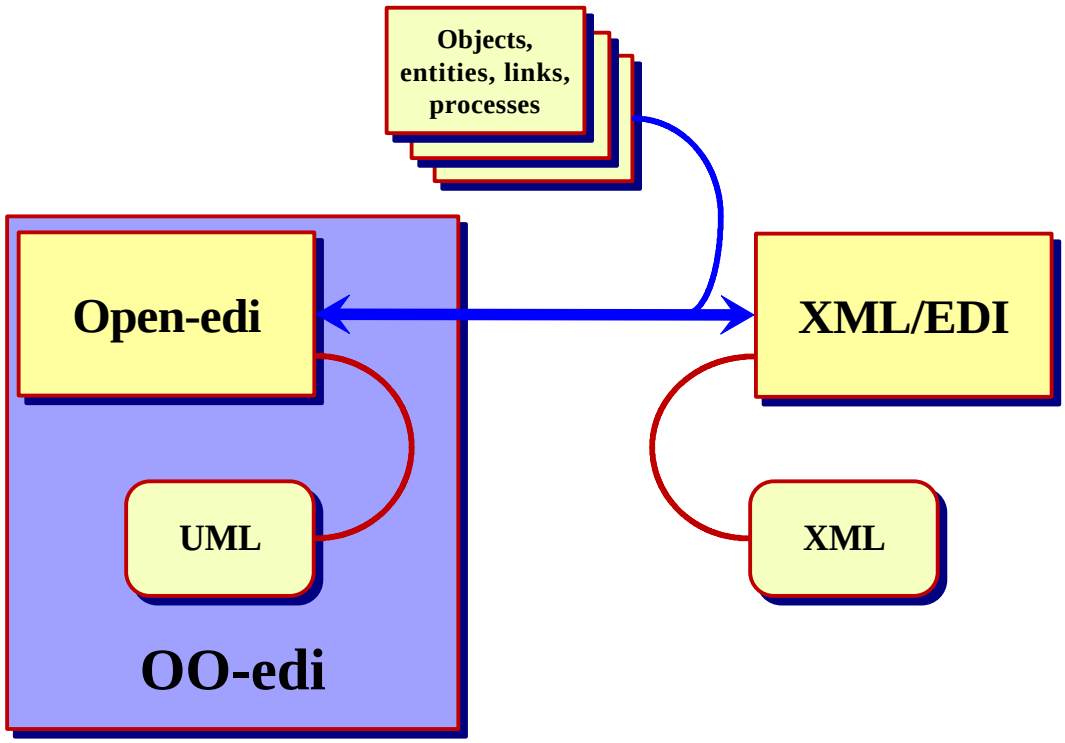

Figure 1: Open-edi and XML/EDI compatibility

[1] The use of a modelling technique, such as UML, to define all aspects of the BOV, is defined by UN/CEFACT as OO-edi (Object Oriented – edi). This work is in progress by the Techniques and Methodology Working Group (TMWG) of CEFACT.

The Coupling of OO-edi and XML/EDI.

When viewed in the context of a logical representation and a physical representation the link between OO-edi and XML/EDI becomes obvious.

However, we now need to consider exactly how the two are equated and the enabling pieces that provide the actual physical linkage. Most importantly we need to show how this is a two way interchange. Two way interchanging means that OO-edi (Open-edi/BOV & UML) can be used to generate a physical XML/EDI system directly, or alternatively, an ad hoc physical system created in XML/EDI can then be imported directly into UML and modelled in as an OO-edi scenario to document the design and verify its accuracy.

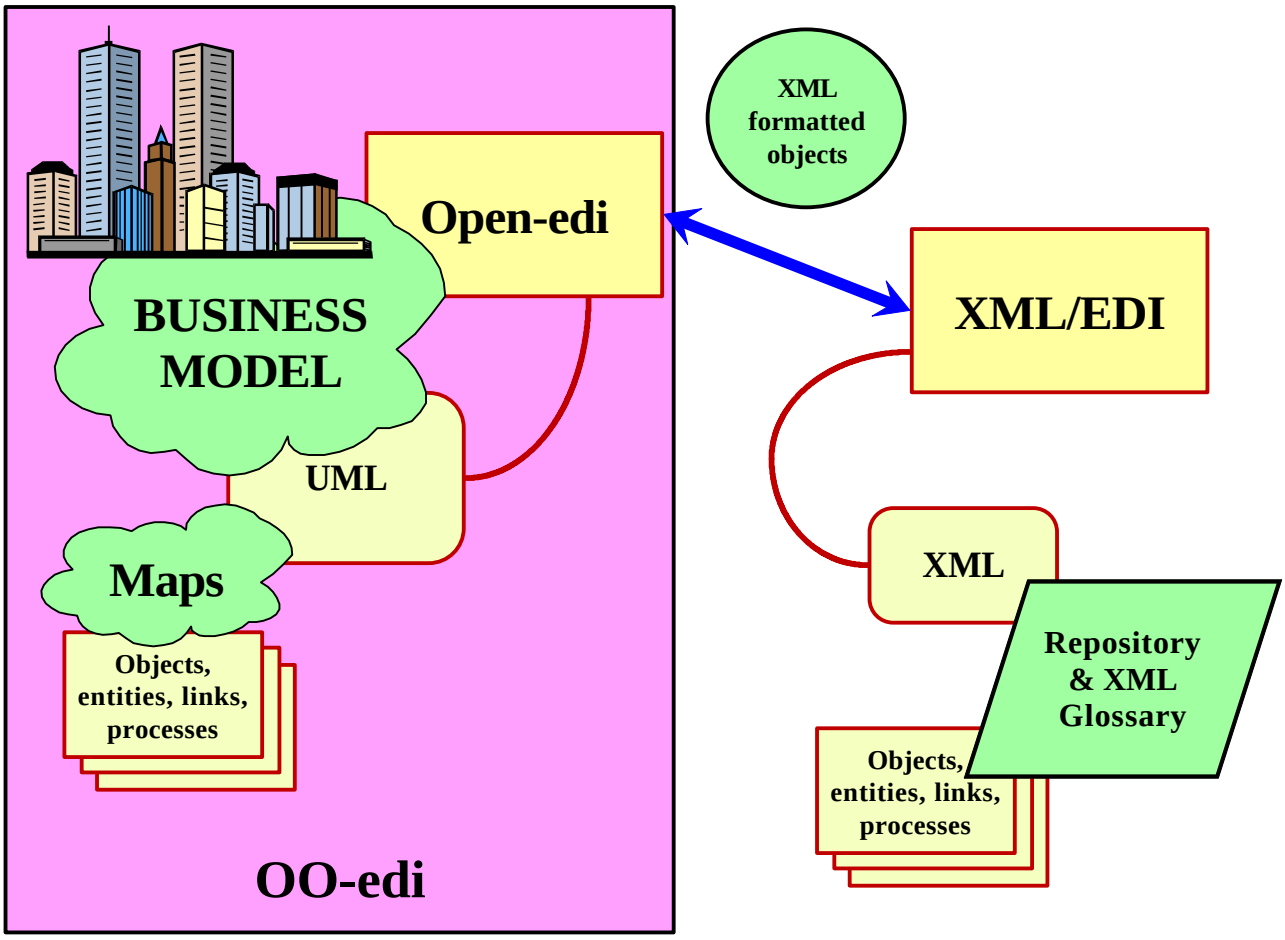

Figure 2: Electronic Business Enablement with OO-edi and XML/EDI

There are several components that enable the linking process and we will next consider these each in turn. Each component provides part of an inter-related whole, and also part of the requirements for XML/EDI deployment.

(More details on each of these components can be found from the Guidelines document available from http://www.xmledi.com).

Open-edi

Open-edi is an ISO/IEC vision of future EDI. ISO/IEC 14662 provides a baseline for all levels of standards that are needed for the specification of Open-edi scenarios and their implementation. TMWG has explored and recommended various modelling techniques including UML for business modelling and information modelling.

Open-edi takes a generic approach. It enables organizations to establish short-term relationships quickly and cost effectively. Open-edi provides the opportunity to lower significantly the barriers to electronic data exchange by introducing standard business scenarios and the necessary services to support them. In principle, once a business scenario is agreed upon, and implementations conform to the Open-edi standards, there is no need for prior agreement among trading partners, other than the decision to engage in the Open-edi transaction in compliance with the business scenario.

The field of application of Open-edi is the electronic processing of business transactions among autonomous multiple organizations within and across sectors (e.g., public, private, industrial, geographic). It includes business transactions that involve multiple data types such as numbers, characters, images and sound. The Open-edi Reference Model provides the standards required for the inter-working of organizations, through interconnected information technology systems, and is independent of specific information technology (IT) implementations, business content or conventions, business activities and organizations.

The Open-edi Reference Model places existing EDI standards in perspective using two views to describe the relevant aspects of business transactions: the Business Operational View (BOV) and the Functional Service View (FSV).

The BOV addresses the aspects of a) the semantics of business data in business transactions and associated data interchanges, and b) the rules for business transactions which apply to the business needs of Open-edi, including:

- operational conventions,
- agreements,
- mutual obligations.

The FSV addresses the supporting services meeting the mechanistic needs of Open-edi. It focuses on the Information Technology aspects of functional capabilities, service interfaces, and protocols, including

- capability of initiating, operating and tracking the progress of Open-edi transactions,
- user application interface,
- transfer infrastructure interface,
- security mechanism handling,
- protocols for inter working of information technology systems of different organizations,
- translation mechanisms.

Figure 3 sets out the relationship between the Open-edi reference model and these views. While the Model itself is developed as a standard, other standards are required. Standards are driven by, and must satisfy, the requirements identified within the Open-edi Reference Model.

Although TMWG's work must satisfy the overall requirements of the Model, the primary focus shall reside with the BOV and shall be independent of the supporting FSV solution. TMWG's assumption is that the FSV will be developed by commercial software vendors which enable distributed object computing environments, and ensure backward compatibility to traditional EDI messages. As such, the resultant BOV related standards will provide the business and object class models to construct Open-edi scenarios.

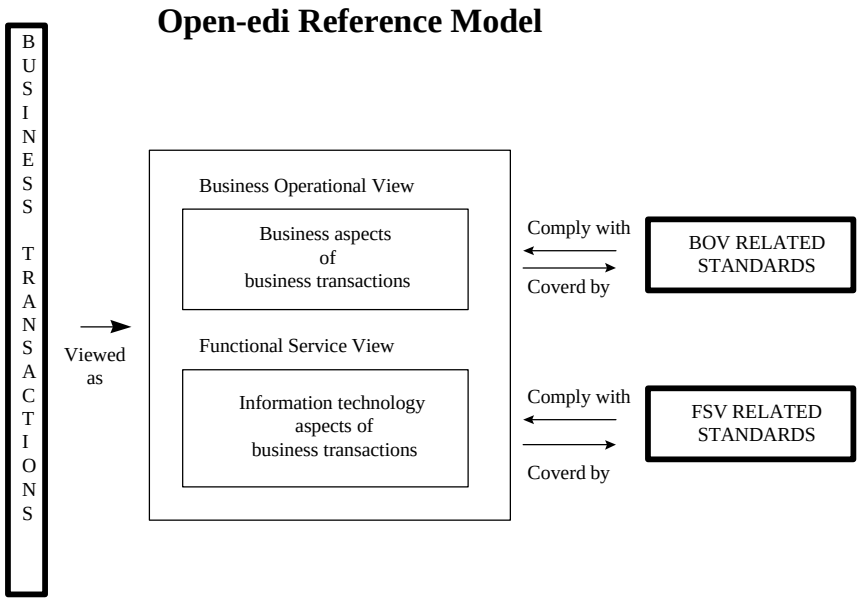

Figure 3 - Open-edi environment

Shift the Focus

Interoperation among application programs requires that there be "common ground" in their exchange of information, so that there can be common understanding and agreement on the information being jointly processed. Common ground in this exchange of information is accomplished in current EDI methodology through a neutral, application independent syntax, i.e., typically for business data a translated UN/EDIFACT interchange file. All consideration of application programs, how to facilitate their interoperation, functionality variations, and the business practices behind them are deliberately ignored. Instead, the current EDI standardization process in UN/EDIFACT concentrates solely on the structure and content of the translated interchange file. The problems associated with UN/EDIFACT standards, and the standard development process, are well documented and are not repeated here.

However, it is essential to understand that for Open-edi to overcome the current impediments to implementing EDI, a new paradigm must be envisioned that shifts the focus on EDI standards from the interchange file to the information contained in the business processes. While business practices from one business organization to another are highly variable, depending on competitive strategies, experience and management style, activities can be decomposed into business processes that are more generic to the type of business. This analysis through the modelling process will identify object classes and models that are likely candidates for standardization. TMWG looks for standard reusable components from which to construct information exchange software. Such a goal is a core concept of object technology.

Modelling the Business Process

As TMWG considers the necessity to decompose business processes to their more generic components, UN/EDIFACT must utilize a consistent methodology (modelling techniques) for conducting the analysis. Thus, it is important to explore the benefits of using modelling techniques to identify the data requirements and data flows of a particular business process. These models assist in providing an interface specification that enables non-standard data, internal to a business process, to be mapped and translated to a representation of standardized data.

It is proposed that models, which provide the interface specification, will constitute the new EDI standards, once they are certified as satisfying the business requirements. These new EDI standards will be independent of the interchange data syntax, transport infrastructure, and EDI server software.

TMWG's primary objective in this new focus on EDI standards is to make EDI technology widely available, non-obtrusive to the business process, and cost effective for all organizations, including SMEs. The requirements to make this objective a reality include:

- Production of well-defined, consistent standards for interoperability, i.e., reduces the number of ways of doing things.
- Development of off-the-shelf tools that are commercially available for analysis and implementation.
- Separation of analysis from application design and programming.
- Availability of training and reference sources (i.e., take advantage of a mainstream methodology for new projects in industry).
- Provision for automatic generation of EDI interactions.
- Separation of data definition and format from the transport layer.

The Healthcare HL7 community is taking a very similar approach in their definition of the RIM (Reference Information Model) model for Healthcare. Their objective is to establish a business model of all the major aspects of Healthcare. Implementors then select the business objects required to fulfil their own processes and implement them as a set of interconnected methods. The HL7 RIM gives an actual example with specific use cases and is therefore very useful in illustrating how this approach works in practice.

For more information on the HL7 RIM, and an example of a model see:
- http://www.mcis.duke.edu/standards/HL7/committees/sgml/WhitePapers/RIM/xml_rim.rtf
- http://www.mcis.duke.edu/standards/HL7/committees/sgml/WhitePapers/RIM/xml_rim.ppt

Unified Modelling Language

TMWG selected Unified Modelling Language (UML) as an integrated suite of modelling techniques to characterize these new EDI standards. Ten UML specification and modelling artifacts are currently under investigation for use in the business process and information modelling methodology as follows:

- Problem statement (per scenario)
- Requirements specification (per scenario)
- Use case specification (per sub-scenario)
- Use case diagram (per scenario)
- Object Class identification and description (using CRC job aid)
- Class diagram (per scenario)
- Activity diagram (per scenario)
- Sequence diagram (per use case)
- Collaboration diagram (per use case)
- Use case state diagram (per use case)

(see http://www.harbinger.com/resource/klaus/tmwg for the latest version [currently 12] of the 'TMWG Reference Guide "The next Generation of UN/EDIFACT" – An Open-edi approach using UML models and OOT –' in order to get a full description of the various components of OO-edi)

Repositories and XML Glossaries

The Repositories in XML/EDI provide the means to define the presentation of existing EDI standards code and element dictionaries, EDI transactions, and XML based documents with Document Type Definitions

(DTD's). Repositories also contain new components that enable the complete definition and representation of a business system in an electronic form. These extras include the following:

- XSL to create GUI form objects from XML;
- XLINK to associate objects and define processes;
- Interchange Transformation Templates (ITT) to capture the EDI Implementation Guideline, and Implementation conventions (IG and IC's) specifics;
- Java, ActiveX, ECMAScript and XFEL (XML Flow Engine Logic) to capture the business processing logic.

These components provide the Functional Service View (FSV) supporting services that meet the mechanistic needs of Open-edi (see details of FSV in earlier Open-edi section of this document).

All these components are captured and described using the XML Glossary. Therefore each repository contains XML Glossaries, and these allow the storage and retrieval of the appropriate components.

An XML Glossary contains the following types of objects:

- Facts
- Categories and sections
- Structures
- Processes
- Events

(See ftp://www.eccnet.com/pub/xmledi/xml-rep.htm for a full description of the Repository approach and design details).

When you review the tools and representation components that UML provides it is clear that by equating XLINK and XML Glossary hierarchy and associations to them a one-to-one equivalence is apparent.

Therefore, UML tools can export into an XML format that can be loaded directly to populate a Repository and vice versa.

In contrasting the different roles the Repositories are specifically designed to include an Applications Programming Interface (API) and the ability to search and retrieve items within repositories from via the internet. UML is designed to document and display the actual design and the relationships between objects and entities.

Process Components - XFEL, Java, ActiveX, ECMAScript

Using Java, ECMAScript or ActiveX allows the development of process objects that can be tied to discrete parts of the Open-edi and UML model. This approach is the same as currently employed with development tools and Java Beans where the design process has access to components that have known behaviours and can therefore be associated and generated into the underlying application processing.

This approach provides a means to attach process components to GUI forms, however it lacks the means to provide the definition of a broad application framework and map directly between UML and XML. That role is fulfilled using the XFEL (XML Flow Engine Logic) technology. (Note: XFEL is described in a separate working paper, and in depth treatment of XFEL is beyond the scope of this paper). To summarize XFEL is designed to allow the use of XML to define processing modules. It achieves this by providing a built-in Flow Engine module that runs in the browser environment and can fully process business application needs in a multi-tier client/server environment. The browser today already contains all the necessary components such as flow objects, JDBC, http transport layer, the Document Object Model, and so on. The Flow Engine processing merely orchestrates these for the application developer. To do this the

XML based XFEL syntax delivers table driven scripts that define the behaviour required. GUI form processing, batch processing and report processing are all defined using a single consistent paradigm. Similarly, menus, user controls, events and task flows can all be specified in table formats that are ideal for XML to express.

The Repository and XML Glossary definitions provide the link between UML and XFEL using XML representations throughout. XFEL lends itself to highly automated Application Wizard processes. Once you have defined in UML the objects, processes and data stores, these can be mapped directly into XML Glossary entities and then XFEL components built from these. An example would be a typical GUI table lookup form with a columnar display format. The Application Wizard can generate any of these given a particular data store, its definition in the XML Glossary and a flow objects form template of the look and feel required.

XFEL can call Java, ECMAScript, ActiveX or even XSL components or vice versa. This provides the means to extend the basic capabilities of the XFEL to handle complex and non-standard processing needs, or provide additional flexibility and customizations.

To summarize combining Java, ECMAScript, ActiveX and XFEL provides the means to deliver application syntax in XML. Therefore XML/EDI delivers the application deployment layer to implement Open-edi and UML directly.

Capabilities Summary

Using the combination of UML and XML as tools we can provide a link between the OO-edi Business Models and XML/EDI deployment.

Figure 4 below shows how this concept works in practice, where the end user is directly interacting with software components that required little or no programming to develop. The business model derived in OO-edi and UML contains all the definitions to allow the generation of appropriate application modules, menus and processing steps and links.

Within the UML tool GUI forms are developed and designed. These are then automatically deployed to HTML flow objects and the end user interface forms.

The Flow Engine Logic contains all the built-in process default knowledge to process a form within the browser environment. Read and writing records and queries and updates are required, and controlling the device interaction with mouse and keyboard input according to the rules specified in the design. Such two way table driven methods have already been demonstrated in two commercially available products. One is a CASE tool that supports UML, the other a table driven 4GL deployment environment that is architecturally very similar to a web browser thin client model (see http://www.popkin.com for details of the Systems Architect Bridge product).

The aim with OO-edi and XML/EDI with XFEL is to provide 80% or more of the application logic with such automated methods. The remaining 20% of discrete logic can be handled with associated process objects developed with tools such as Java. UML provides excellent coupling to object deployment environments such a Java, so these can be freely included and re-used consistently through the XML/EDI application space.

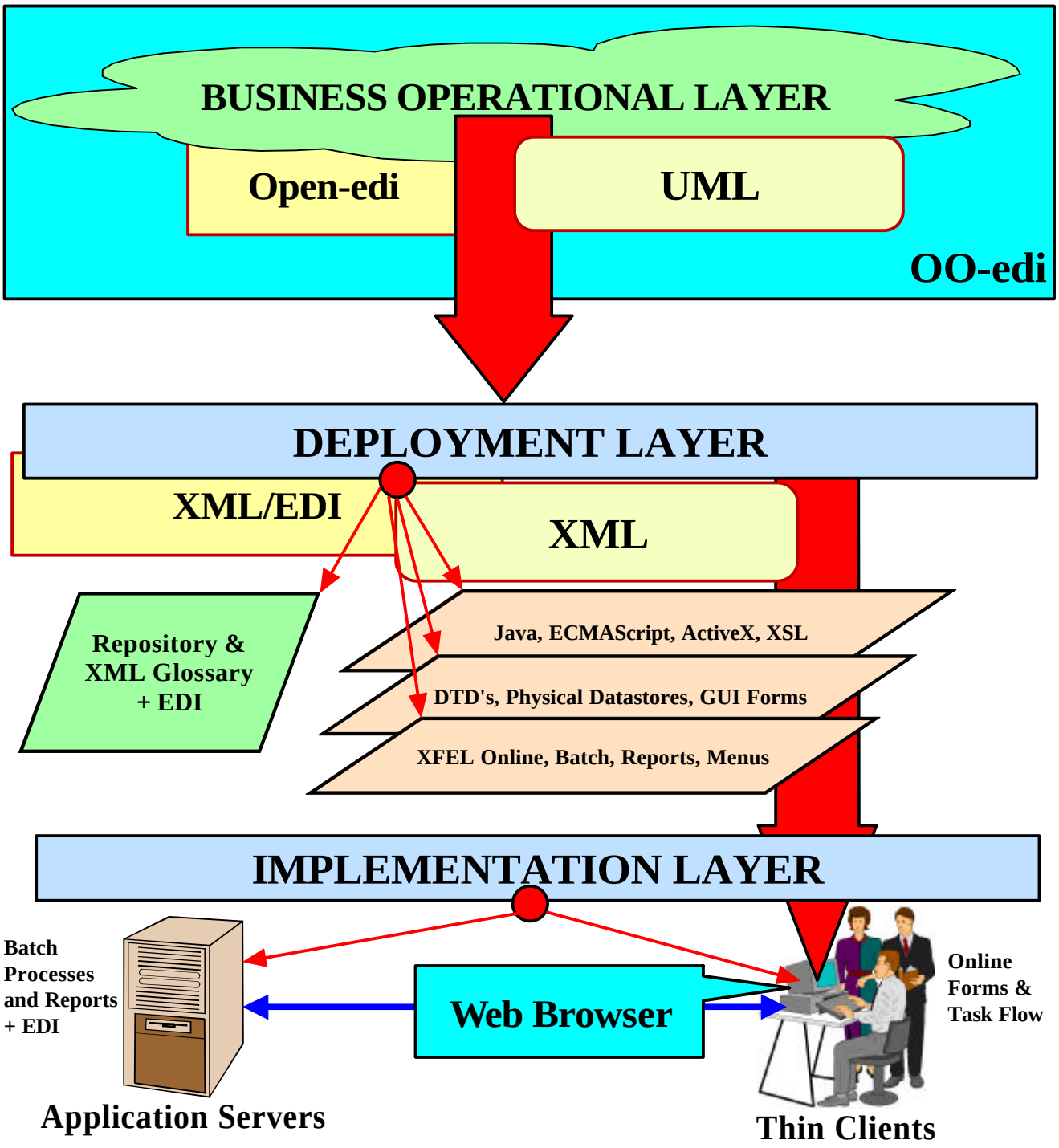

Figure 4: OO-edi design and XML/EDI deployment

Notes:

The HL7 HealthCare RIM (Reference Information Model) available at

http://www.mcis.duke.edu/standards/HL7/committees/sgml/WhitePapers/RIM/xml_rim.rtf

shows an existing example of marrying an business architectural model to an XML representation. (An area for further work is to explore alignment between OO-edi and HL7 RIM).

A draft working document on XFEL is currently under review and will be released for public comment within the next few weeks.

Addendum:

Open-edi –Reference Model is available from: http://www.iso.ch (under JTC1/SC32/WG1)

Further information on work on integrating UML and XML can be found at:
http://www.marketplace.unisys.com/products/urep/ and from:
http://www.yy.cs.keio.ac.jp/~suzuki/project/uxf/

The following two figures are taken from a presentation by Scott Nieman. The full presentation is available from: http://www.xmledi.com/x12/uml-xml.ppt.

A related discussion documents on modelling approaches can be found at:
http://www.erols.com/roebuckr/ue-gem/ and at http://www.imsproject.org/technical/metadata/

Figure A1 - UML Representation

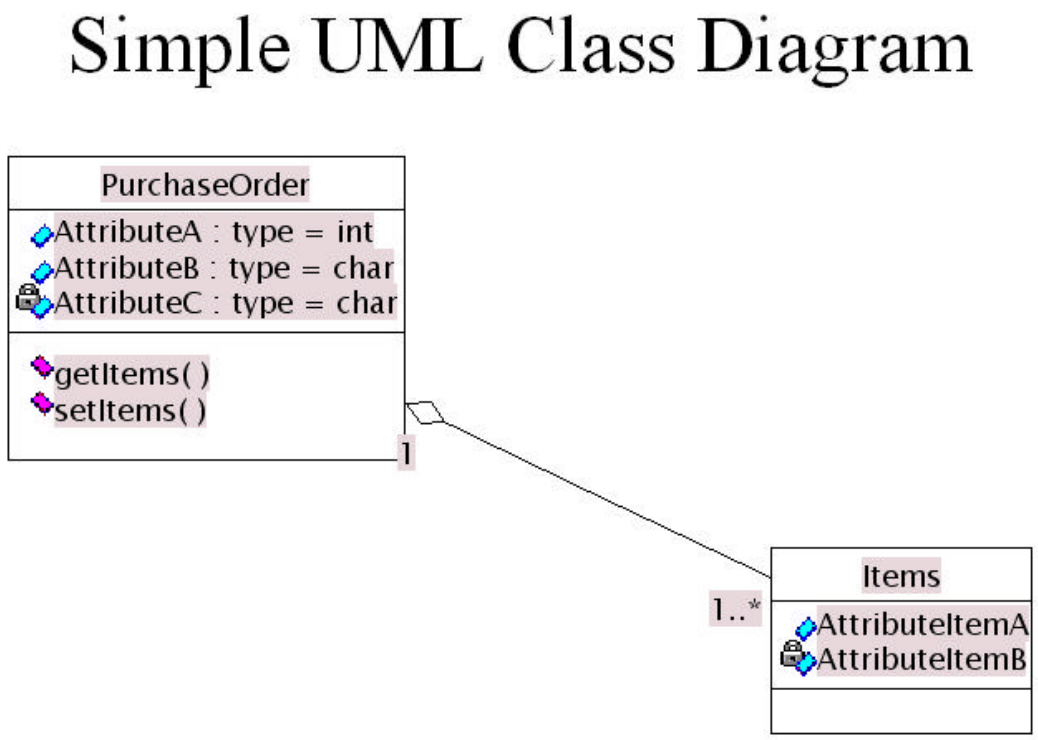

Figure A2 - XML Equivalent

Resulting XML MetaData Document

```
<Class> <ClassName>PurchaseOrder</ClassName>
     <Attribute>PurchaseOrder.AttributeA</Attribute>
     <Attribute>PurchaseOrder.AttributeB</Attribute>
     <Attribute>PurchaseOrder.AttributeC</Attribute>
     <Operation>PurchaseOrder.getItems</Operation>
     <Operation>PurchaseOrder.setItems</Operation>
</Class>
<Class> <ClassName>Item</ClassName>
     <Attribute>Item.AttributeItemA</Attribute>
     <Attribute>Item.AttributeItemB</Attribute>
</Class>
```