e-Everything: e-Commerce, e-Government, e-Household, e-Democracy

14th Bled Electronic Commerce Conference

Bled, Slovenia, June 25 - 26, 2001

A Graphical Design Interface for XML Schemas

Ronald M. Lee

Erasmus University Research Institute for Decision and Information Systems (EURIDIS)
Erasmus University Rotterdam, P.O. BOX 1738/F1-27, Rotterdam, The Netherlands
R.Lee@euridis.fbk.eur.nl

Abstract

A graphical interface for designing XML Schemas is presented. The current version of this interface is based on the Schema for Object Oriented XML (SOX), but the constructs are easily generalizable for other schema representations as well. The main advantage is to use the two-dimensional visual field of graphical interface plus multiple windowing to capture more intuitively represent the hierarchical decomposition and re-usability of the XML element definitions. Initial user experience has been quite positive.

Keywords: *XML; XML Schemas; Schema for Object-Oriented XML (SOX); W3C XML Schema Language*

1. Introduction: XML Schemas

Extensible Markup Language, XML, is growing in popularity as the preferred representation for the exchange of structured business data and documents [1, 5, 7, 8, 9, 14]. Until recently, the most common computer format for exchanging business data has been Electronic Data Interchange, EDI, most notably UN/EDIFACT [21, 22]. A principal advantage of XML over EDI is that the structure of the XML document is in a standardized syntax that is machine interpretable. This allows the development of generic software that can process any variety of XML documents.

The basic syntax for XML data structures is the XML Document Type Definitions (DTD) language. XML/DTD is a direct adaptation of the parent SGML/DTD, whose target was mainly the structuring of textual documents [17]. However, the

XML/DTD is not sufficiently perspicuous for many business data applications. For this reason, the notion of XML Schemas is being developed [6, 7, 11, 24]. The main contribution of an XML Schema language is to add stronger data typing than is provided by the DTD. (Actually, the XML Schema language is defined by a DTD, so the schema itself is in the XML document syntax.) Another contribution of XML schemas is the re-usability of data structures. For example, a purchase order document may contain fields for delivery address and billing address. These in turn contain fields for street, city, postal zone, which are reused in each address structure definition. Furthermore, the syntax for data items such as postal zone vary from one country to another, and can be given specific, reusable datatype definitions within the XML Schema language. These kinds of re-usable structures can often become fairly complex, with numerous levels of embedding.

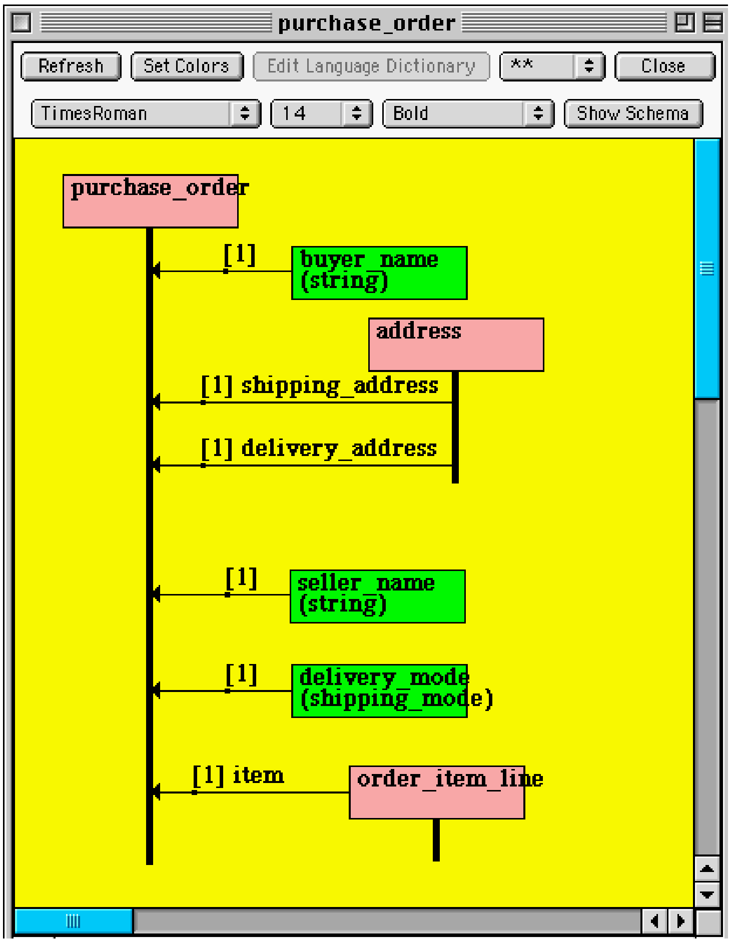


Figure 1: *Graphical Design of Element Defintion: Purchase Order*

XML Schemas can be defined and edited using a simple text editor. However, since the embedded structure definitions can often become complex, specialized editors can be helpful. Since XML Schemas are expressed in XML syntax, XML editing tools can be used. These aid in checking that the XML Schema is syntactically correct. Nonetheless, especially for less technical users, the linear syntax of these editors often makes it difficult to visualize the data structure. In this paper we propose a graphical interface for editing XML Schemas. It is based on the graphical style of the class diagrams of the Unified Modeling Language (UML). A somewhat similar use of class diagrams is also proposed in [19]. As will be discussed further below, there are several candidate representations for XML Schemas now available. The one used here is the SOX (Schema for Object-oriented XML), developed by CommerceOne [6, 25]. The graphical interface design could easily be adapted for other representations, for example that of the W3C [25], when it is finally approved as a standard.

2. Graphical Design Interface for XML Schemas

The XML Schema interface described here was originally developed as part of a larger modeling system known as InterProcs [12, 13]. However, since these aspects are relatively separate from the rest, there is now a separate tool, called Case-XML, containing only the XML editing aspects. See [10] for more details and access to executable examples and software. The software was written in Java. We will illustrate the graphical interface with an example document, purchase_order, as shown in Figure 1. Briefly, the boxes each represent an element definition, which may be either basic or complex. Complex elements may have further graphical structure, for instance, order_item_line, as shown in Figure 2. The notation here borrows from UML Class Diagrams, which in turn have similarities to earlier Entity-Relationship diagrams [3], and data structure diagrams [4]. A graphical novelty here is the "tail" emerging from the complex element defintions. This allows the sequence of the component entity definitions to be specified graphically. The various graphical tools of this interface are identified in the menu snapshot shown in Figure 3. These tools are reasonably self-explanatory.

Whereas complex element types are defined by other schema graphs, basic element types are defined using a structured dialog box, shown in Figure 4. Here the datatype of the basic element type is selected, which is either pre-defined ('intrinsic'), or defined by the user. An example of the latter is an Enumeration, as illustrated in Figure 5, for the data item 'delivery_mode'. This is another dialog box, used to specify the various choices.

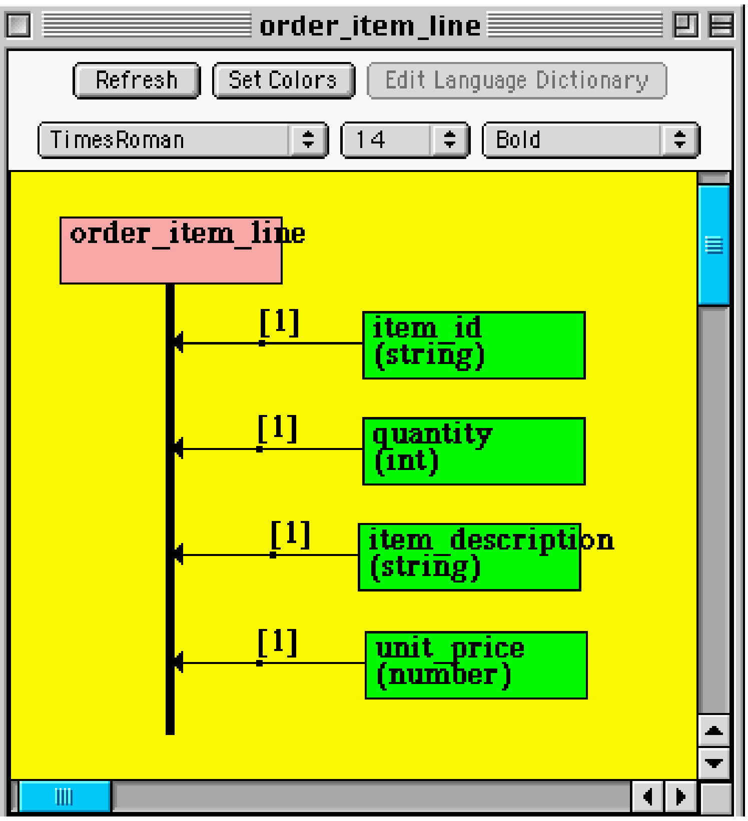


Figure 2: *Graphical Design of Element Definition: Order Item Line*

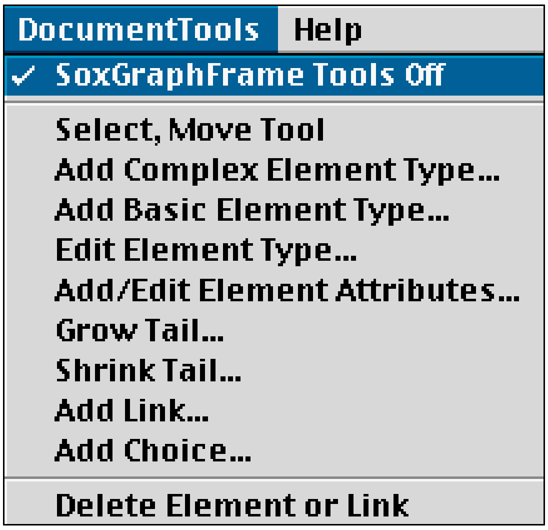
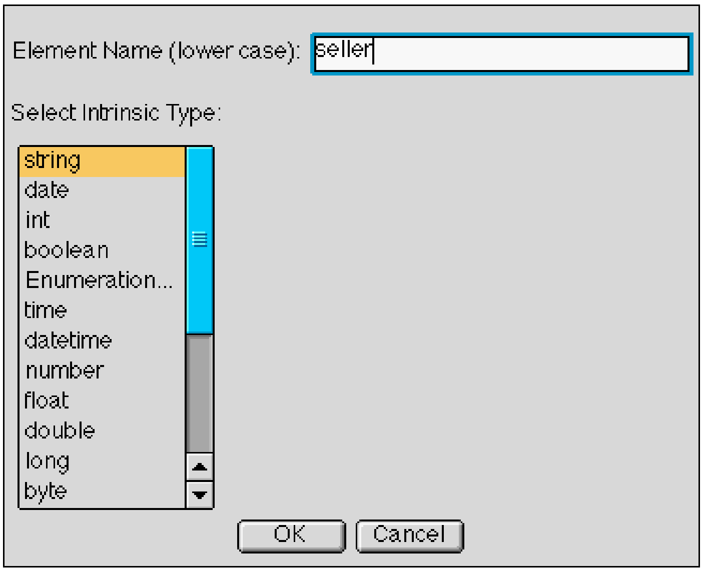


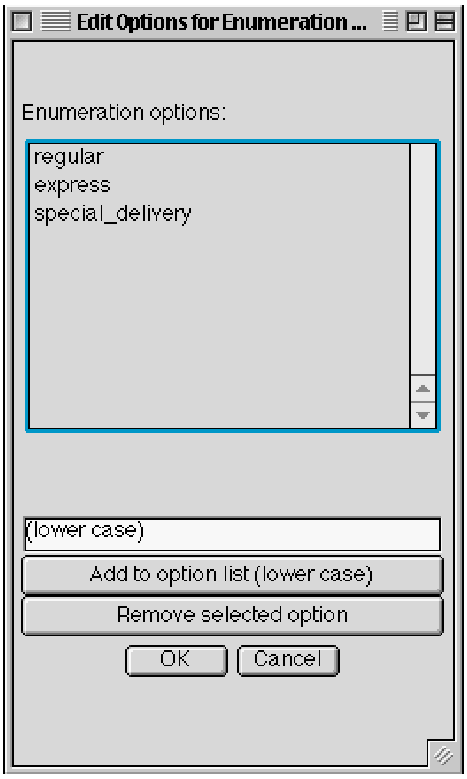
Figure 3: *Graphical Tool Menu for XML Schemas*



Element Name (lower case): seller

Select Intrinsic Type: string

Figure 4: *Dialog to Select Datatype*



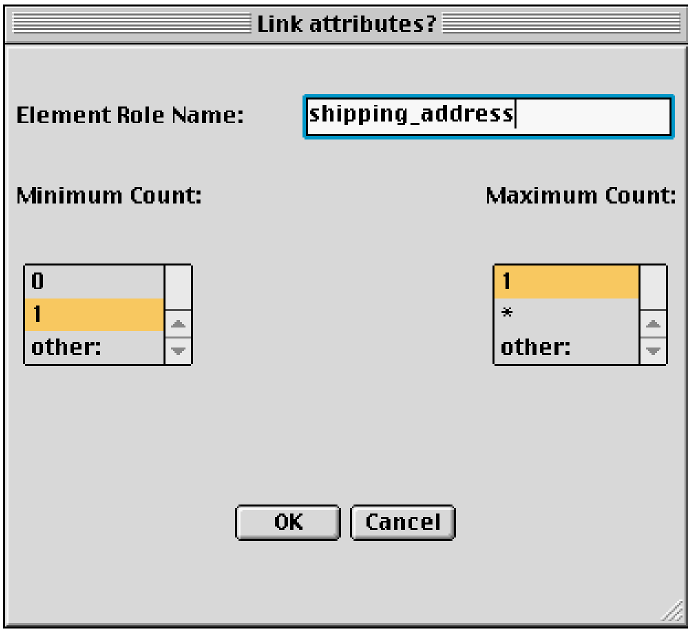
Edit Options for Enumeration ...

Enumeration options:

regular
express
special_delivery

Figure 5: *Dialog Box for Enumeration of Choices*

The decomposition of a complex element type is done by means of 'links' from the component element types, which may be either other complex element types, or basic element types. In both of these cases, a cardinality is specified, indicating the minimum and maximum counts of the components. Typically this is [1] indicating that exactly one component is required. The cardinality [0,1] indicates that it is optional, and [0, *] and [1,*] indicate that zero-many or one-many are required, respectively. Additionally, the link connecting from a complex component element type may specify a 'role' name, which enables the complex type to be re-used several times in a single definition. The re-use of 'address' as 'shipping_address" and 'billing_address' illustrates this. Such a link specification is shown in Figure 6.



Element Role Name: shipping_address

Minimum Count:

0
1
other:

Maximum Count:

1
*
other:

Figure 6: *Dialog to Specify Link from Complex Element*

As is common in graphical interfaces, the Select, Move Tool may be used to rearrange the graph elements in the drawing area.

3. Generation of XML Schema Syntax

Once the graphical diagram has been completed, the button "Show Schema" will generate the standard XML Schema syntax. Figure 7 shows the generated XML Schema syntax for the diagrams shown above.

```
<?xml version="1.0"?>
<!DOCTYPE schema SYSTEM "urn:x-:xdk:xml:schema.dtd$1.0">
<schema uri="urn:x- commerceone:CBL:CBL.sox$1.0">
<elementtype name="purchase_order">
<model>
        <sequence>
            <element type="string" name="buyer_name" />
            <element type="address" />
            <element type="address" name="shipping_address" />
            <element type="address" name="delivery_address" />
            <element type="string" name="seller_name" />
            <element type="shipping_mode" name="delivery_mode" />
            <element type="order_item_line" name="item" />
        </sequence>
</model>
</elementtype>
<elementtype name="order_item_line">
<model>
        <sequence>
            <element type="string" name="item_id" />
            <element type="int" name="quantity" />
            <element type="string" name="item_description" />
            <element type="number" name="unit_price" />
        </sequence>
</model>
</elementtype>
<elementtype name="address">
<model>
        <sequence>
            <element type="string" name="address_line_1" />
            <element type="string" name="address_line_2" />
            <element type="string" name="city_name" />
            <element type="string" name="postal_code" />
            <element type="string" name="country_name" />
        </sequence>
</model>
</elementtype>
<datatype name="shipping_mode">
    <enumeration datatype= "string">
        <option>regular</option>
        <option>express</option>
        <option>special_delivery</option>
    </enumeration>
</datatype>
</schema>
```

Figure 7: *Generated XML Schema Syntax*

4. Electronic Document Layout

XML Schemas are of course for use for sending and receiving actual electronic documents. For this, a screen layout of the document is needed. The button "Show Layout" provides a simple electronic document layout corresponding to the defined XML Schema. (The software also includes sending and receiving of documents.) An example is shown in Figure 8 (a,b).

However, especially for business documents used in international trade, a more sophisticated layout is warranted. For example the United Nations has developed a "layout key" that specifies the physical layout of most international trade documents in such a way that parties in with different native languages can understand the fields of the document by their location on the form [20]. This UN layout key was originally designed for paper documents, but currently is being applied for electronic documents in the Electra project of SITPRO (UK) [18]. Electra is also being extended to accomodate XML document definitions, and so could in principle use the results of this design tool.

5. Other Schema Languages

In October 2000 the XML Schema 1.0 language became a candidate recommendation at the World Wide Web Consortium (W3C). Besides this W3C XML Schema language, several other schema proposals exist. The Robin Cover [7] offers an extensive and up-to-date overview of the most important schema languages:

XML Schema Definition Language (XML Schema)

XML-Data Reduced (XDR)

Document Content Description (DCD)

Schema for Object-oriented XML (SOX)

Schematron

Datatypes for DTDs (DT4DTD)

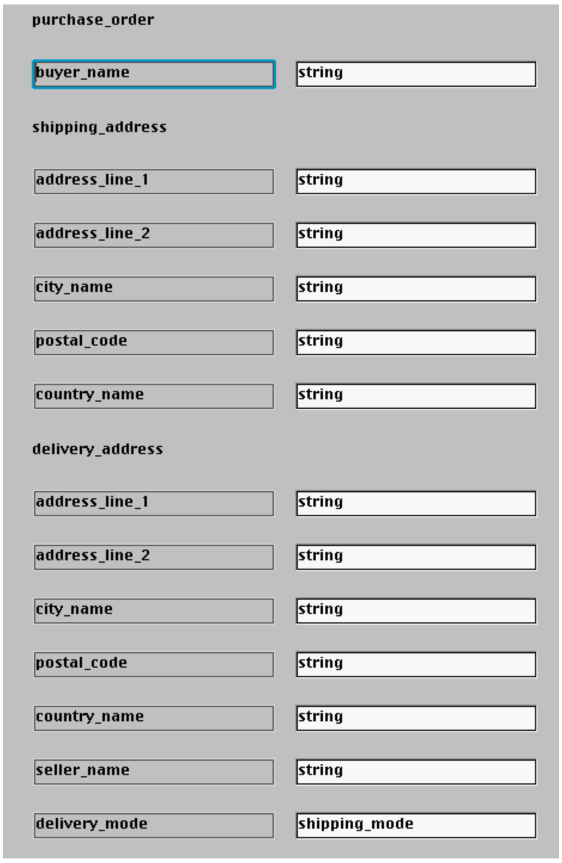
REgular LAnguage description for XML (RELAX)

Document Structure Description (DSD)

Lee and Chu [11] offer a comparison between several of the most popular schema languages that are backed by substantial organisations so that their chances of survival is high and for which there is publicly known usage or applications.

XDR. First known as XML-Data, then later trimmed and improved to XDR (XML-Data Reduced), this language is a joint effort of Microsoft and others and is being used in Microsoft's BizTalk framework. XDR is heavily influenced by another proposal co-developed by IBM and Microsoft, DCD (Document Content Description), and thus shares many similar features.

SOX (Schema for Object-Oriented XML) is an alternative schema language for defining the syntactic structure and partial semantics of XML document types. As the name implies, it extends DTD in an object-oriented way by allowing extensible data types and inheritance among element types. The current version, 2.0, is being developed by Commerce One.

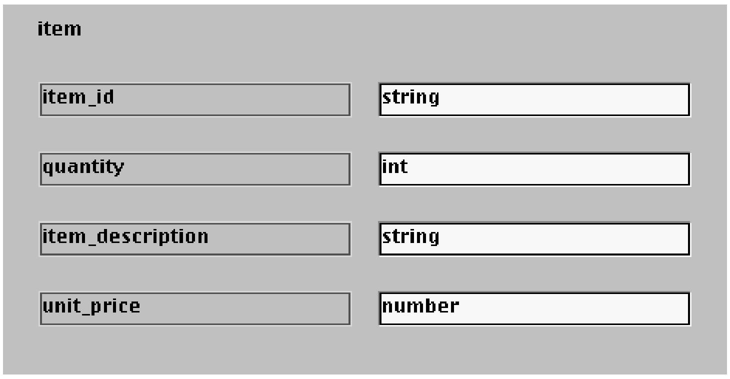


purchase_order	
buyer_name	string
shipping_address	
address_line_1	string
address_line_2	string
city_name	string
postal_code	string
country_name	string
delivery_address	
address_line_1	string
address_line_2	string
city_name	string
postal_code	string
country_name	string
seller_name	string
delivery_mode	shipping_mode

Figure 8a: *Sample Screen Layout for Electronic Document*

Schematron, created by Rick Jelliffe, is quite unique from others in that it focuses on validating schemas using patterns instead of defining schemas. Its schema definition is simple enough to be defined in a single page, yet provides very powerful constraint specification via a language called XPath. The latest version is 1.4.

DSD 1.0 was co-developed by AT&T Labs and BRICS with the goals of context-dependent description of elements and attributes, flexible default insertion mechanisms, expressive power close to XSLT etc. Like Schematron, DSD makes strong use of schema constraints.



item

item_id	string
quantity	int
item_description	string
unit_price	number

Figure 8b: *Sample Screen Layout for Electronic Document (continued)*

According to Lee and Chu [11], the expressive power of the schema languages can be ranked in three levels, as shown in Figure 9.

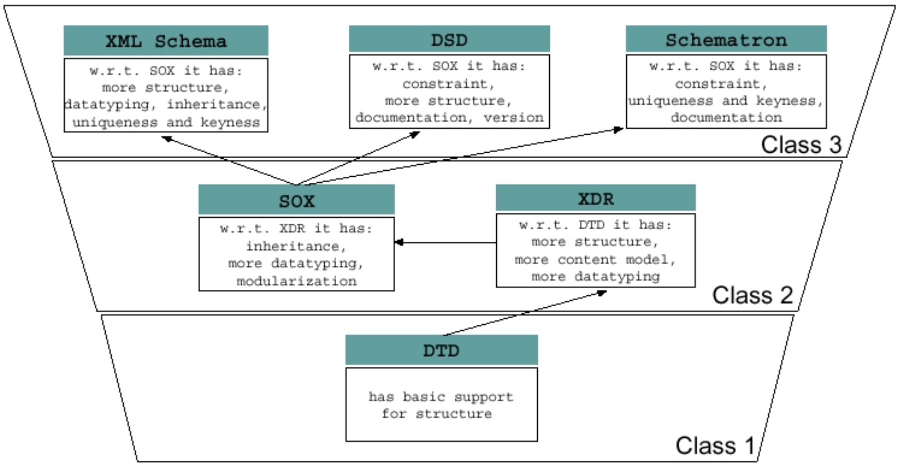


Figure 9: *Classification of the expressive power of various schema languages (from Lee and Chu, [11])*

At the lowest level is the XML DTD, which lacks both data typing and other constraints. The middle level includes schema languages with data typing, but not constraints. The SOX schema language, adopted for this paper, is at this level. The top level includes not only data typing, but also schema constraints. The W3C Schema Language is included here. The XML Schema language of W3C (currently a candidate recommendation) is most likely to become the worldwide standard for schema languages. This is because it has strong expressive power and the W3C is widely recognised by the industry.

6. Concluding Remarks

A graphical interface for designing XML Schemas has been presented. The current version is based on the Schema for Object Oriented XML (SOX), but can be adapted for other schema representations as well. This graphical interface captures mainly the hierarchical decomposition and re-usability of the XML element definitions. The detail data typing is specified using the underlying schema language vocabulary. Likewise, if and when constraint specifications are included, these too will likely be done in the syntax of the underlying schema language, rather than graphically.

Initial experience (with students) indicates that the graphical interface is relatively easy and fun to learn, and certainly is far less intimidating than the schema language syntax.

Acknowledgments: thanks to Walter Thoen and Menno de Vreede for their useful comments and suggestions about this graphical interface.

References

[1] BizTalk, http://www.BizTalk.org

[2] Booch, G., Rumbaugh, J. & Jacobson, I. (1999). The Unified Modeling Language User Guide. Reading, MA: Addison-Wesley.

[3] Chen, Peter. "the Entity-Relationship Model -- Toward a Unified View of Data", ACM Transactions on Database Systems, 1, 1976: 9-36.

[4] CODASYL, Data Base Task Group (DBTG) Report. ACM, 1971.

[5] CommerceNet http://www.commerce.net

[6] CommerceOne: "Sox Tutorial"

www.commerceone.com/xml/cbl/index.html

[7] Cover, Robin. "The XML Cover Pages", http://www.oasis-open.org/cover/

[8] ebXML, http://www.ebxml.org

[9] GE Information Services (1999), XML's Impact on E-commerce, a closer look at its role in EDI, EAI and Supply-Chain Processes, available at: http://www.gegxs.com/geiscom/downloads/xmlwhitepaper.pdf

[10] INTERPROCS, http://abduction.euridis.fbk.eur.nl/projects/InterProcs.html

[11] Lee, Dongwon, Wesley W. Chu, "Comparative Analysis of Six XML Schema Languages", ACM SIGMOD Record Volume 29, Number 3 (September, 2000).

[12] Lee, R.M. "INTERPROCS: A Java-based Prototyping Environment for Distributed Electronic Trade Procedures", Proceedings of the Hawaii International Conference on System Sciences, January, 1998, pp. 202-209.

[13] Lee, R.M. "Distributed Electronic Trade Scenarios: Representation, Design, Prototyping", International Journal on Electronic Commerce : Special Issue on Formal Aspects of Digital Commerce, eds. S. O. Kimbough and R.M. Lee, Vol 3, No 1, 1999, pp. 23-26.

[14] OASIS, http://www.oasis-open.org

[15] Open Buying on the Internet, http://www.openbuy.org

[16] RosettaNet http://www.rosettanet.org

[17] SGML/XML Users Group http://www.sgml-ug.nl

[18] SITPRO, ElecTra, www.sitpro.org.uk

[19] SWIFT, "From UML to XML", Internal Report, 2000.

[20] UN/CEFACT, "Application of the United Nations Layout Key: Guidelines and Examples", TRACE/CEFACT/2001/16, available at: http://www.unece.org/cefact/

[21] UN/EDIFACT, http://www.edifact-wg.org

[22] United Nations (1993), ISO CD 9735: 1993(E), Electronic Data Interchange for Administration, Commerce and Transport (EDIFACT) – Application level syntax rules, 1993.

[23] Walker, Laura (2000), OASIS: The latest on XML and its role in Electronic Business

[24] W3C, "Schema for Object-Oriented XML"

www.w3.org/TR/NOTE-SOX

www.w3.org/TR/xmlschema-0

www.w3.org/TR/xmlschema-1

www.w3.org/TR/xmlschema-2

[25] Xcbl http://www.commerceone.com/xml/cbl

[26] XEDI: Approach to XML - EDI http://www.xedi.org

[27] XML/EDI Group http://www.xmledi.net/

[28] XML Software http://www.xmlsoftware.com