XML-based Supply Chain Management
– as SIMPLEX as it is –

Peter Buxmann, Luis Martín Díaz, Erik Wüstner
Freiberg University of Technology
Chair of Information Systems / Wirtschaftsinformatik
Lessingstr. 45
09596 Freiberg / Germany
URL: http://www.bwl.tu-freiberg.de/wi
E-Mail: {buxmann|luis.martin|erik.wuestner}@bwl.tu-freiberg.de

Abstract

In this paper we want to examine, to what extent XML is able to support the exchange of business documents in supply chains. Thereby we focus on the problem of converting different data formats of participants of the supply chain. First results show that XML and its surrounding standards of the XML family highly accelerate and simplify the conversion process. Therefore, XML allows using a common standard on a lower level, without reducing variety on a higher level, due to the use of different XML vocabularies.

First, we examine different approaches for solving the transformation problem. Second, we show how XML can actually be implemented for a Web-based integration in supply chains. We present a java-based prototype that enables document exchange over the Internet using XML business vocabularies for document representation, XSLT for document conversion and presentation, and both DOM and SAX for processing and integrating documents into in-house-systems.

1. Introduction

The problem of conversion is well known. Common examples are the exchange of files between applications of different software vendors and the incorporation of legacy business data into a new standard software. A similar problem occurs in the field of Supply Chain Management, since participants often use different data formats.

Meanwhile there is a standard available, XML, which is on the way of becoming the universal data format for the Web. In addition, the so-called XML-family provides a variety of complementary standards, which, for example, enable the processing of data marked up in XML.

In this work we examine, to what extent XML is able to support the exchange of business documents in supply chains. Thereby, we concentrate on the problem of converting different data. For this purpose, we examine different approaches for solving the transformation problem.

In the following section we describe the status quo of classical document exchange in supply chains. Then we continue with the role of XML and give an overview of XML standardization initiatives relevant to Supply Chain Management. In the fourth section, we present a prototype called SIMPLEX for displaying, describing, exchanging, converting, and integrating XML-documents. In this section, we also take a closer look at strategies for transforming different data formats. This paper ends with a summary of the implementation experiences and a short outlook on further research.

2. EDI in Supply Chain Management

Supply Chain Management describes the integration beyond the boundaries of the firm, involving business partners in different processes and activities that produce value in the form of products and services [4][12][6][1]. The participants of the supply chain are suppliers, producers, retailers, logistics-service providers, and customers. The main idea is that cooperative planning generally leads to better results than isolated planning, e. g. cost savings.

These better results can be achieved by coordinating procurement, transport, and storage strategies as well as by integrating the planning of demand, production, and distribution.

The usage of information and communication technology plays a key role in Supply Chain
Management. The main goal is to provide information along the supply chain in the most inexpensive and fastest way possible. This can be achieved by using Electronic Data Interchange (EDI) [5], which enables partners to electronically exchange business documents such as forecasts, orders, and delivery notifications.

Industry and nation-specific requirements on a business vocabulary have resulted in the development of many different incompatible EDI standards. The agreement on the usage of one EDI standard among business partners results in a cost reduction, leads to time savings, and allows a less capital binding just-in-time-production [5].

Despite its large savings potential, the use of EDI is not nearly as widespread as one would expect. It is estimated that only about 5 percent of the companies, which could profit from EDI actually use it [11]. The main reason is that small and medium-sized enterprises try to avoid the considerable set-up and operating costs of traditional EDI solutions. Therefore, the use of EDI is mainly reserved for large companies, which, due to the bigger transaction volume, profit more from the improvement of their business processes. According to an empirical analysis, 52 percent of the German and 75 percent of the American large companies use EDI [18]. Yet, 98 percent of non-Fortune 1000 companies do not use it. This can be observed in the automobile supply chain, where the production company often has an EDI connection with its largest suppliers, whereas the small ones are not part of this EDI network.

As far as traditional EDI is concerned, the most cost intensive factor is data transport through VANs (Value Added Networks). This is one important reason why companies switch to the Internet as a medium of data transport. While the costs of using VAN connections are often determined either by the number of messages sent or by the amount of transmitted characters, there is no such calculation when using the Internet. The advantage of the Internet for its users is so immense that 3Com, for instance, expects to handle almost 100% of all EDI traffic over the Internet within the next years [18]. In the following section, we focus on the question whether XML can replace existing EDI solutions. The consequence of this could be the integration of those participants of the supply chain, who could not afford joining traditional EDI networks, yet.

### 3. XML and Supply Chain Management

The Extensible Markup Language (XML) was developed by the World Wide Web Consortium (W3C). It was published as a recommendation in February 1998 [16]. The W3C defines XML as a “data format for structured document interchange on the Web” [15].

The core principle is the strict separation of content and structure of the stored data from its layout [2]. Thus, meaningful data and document structures can be described in an application and vendor-neutral manner. With the help of Document Type Definitions (DTD’s) and schemas, this data can be validated. Furthermore, XML enables flexible processing and simple information exchange between heterogeneous applications, media, and platforms. The presentation and view of an XML-document can be customized for any needs by using technologies such as CSS and XSL [17].

A major factor of XML’s success is that the W3C has created more than just a data format. The XML-family rather consists of a variety of related standards, which enable the processing of data marked up with XML. Besides DTD’s, schemas and style sheets the W3C has developed models for extended linkage (XLink, XPointer) and XML namespaces for a decentralized handling of semantic conflicts. The Document Object Model (DOM), as a standardized model for accessing structured data and an interface for the further processing of XML-documents, has also gained widespread acceptance.

The general difficulty of the integration and standardization of documents, however, is not automatically solved by XML. As long as the supply chain partners do not agree on the tags, patterns and schemas to use, neither a correct interpretation nor a subsequent processing of the documents is possible. Thus, similar to traditional EDI a semantic standardization of message types and data structures is necessary. For this purpose, it makes sense to use available knowledge from conventional EDI.

Therefore, standardization initiatives such as xCBL (XML Common Business Library, http://www.xcbl.org) from Commerce One, the CEN/ISSS XML-EDI pilot project (http://www.cenorm.be/issss/workshop/ec/xmledi/default.html), or the XML/EDI Group (http://www.xmledi.group.org) are based upon EDIFACT or X12-structures.

These initiatives specify how to define elements like addresses, prices, discounts or currencies with XML syntax. These document types are available as DTD, XDR (the Microsoft schema language) and as W3C-conform XML-schema.

Other examples of this approach, which focus on documents and document types, are Ariba’s cXML (Commerce XML, http://www.cxml.org), eBIS-XML (http://www.ebis-xml.net) from BASDA (Business and Accounting Software Developers Association), and OAGIS (Open Applications Group Integration Specification) from the Open Applications Group (http://www.openapplications.org).

In fall 2000 a study identified a total of 250 different e-business vocabularies [7]. The research was focused on the directories XML.com, OASIS/Robin Cover,
Schema.Net and IBM’s alphaWorks as well as on vocabularies, which are registered or maintained at XML.org (OASIS), Microsoft (BizTalk), and DISA. The following figure shows an excerpt from a purchase order tagged in the xCBL 3.0 format:

```xml
<?xml version="1.0" ?>
<Order>
  <OrderHeader>
    <OrderNumber>PO23540586</OrderNumber>
    <SellerOrderNumber>12949457S4</SellerOrderNumber>
    <ListOfMessageID>
      <MessageID>
        <IDNumber>Signed Purchase Order 12335</IDNumber>
        <IDAssignedBy>
          <IDAssignedByCoded>Other</IDAssignedByCoded>
          <IDAssignedByCodedOther>LaMu2386</IDAssignedByCodedOther>
        </IDAssignedBy>
        <IDAssignedDate>20010211T15:25:00</IDAssignedDate>
      </MessageID>
    </ListOfMessageID>
  </OrderNumber>
  <OrderIssueDate>20010225T09:30:00</OrderIssueDate>
  <OrderReferences>
    <AccountCode>
      <Reference>
        <RefNum>58395776904</RefNum>
        <RefDate>20010211T15:26:00</RefDate>
      </Reference>
    </AccountCode>
  </OrderReferences>
</Order>
```

Figure 1. Excerpt from a purchase order in the xCBL 3.0 Format

In addition, a set of XML frameworks is being developed. Examples include Microsoft’s BizTalk (http://www.biztalk.org), ebXML (http://www.ebxml.org) and RosettaNet (http://www.rosettanet.org). These frameworks offer a basis for the structured document exchange between different partners both within and cross industries. They focus on the description of the entire communication process and not just pure content specifications of certain message types.

4. SIMPLEX – Supply Chain Management Platform Enabled by XML

With SIMPLEX (Supply Chain Management Platform Enabled by XML), we are developing an open application for enabling a straightforward exchange of business documents between the partners in the supply chain.

SIMPLEX uses XML to describe and to structure the documents we briefly addressed in the second section. Based on these XML coded documents, SIMPLEX supports the execution of get and put information processes, the conversion between different XML vocabularies, and the integration into in-house-systems.

4.1. The SIMPLEX architecture

SIMPLEX is mainly based upon open standards, open source software, and freeware. The use of the commercial XML database Tamino is the only exception to this principle in SIMPLEX’ architecture. Tamino is currently used for two particular reasons:

- Tamino offers interfaces for integrating XML data in external non-XML data sources such as relational databases.
- XML data stored in Tamino is accessible via a web server. This makes the data available at any time with no other help than an XML-capable browser.

Meanwhile, there are some open source and freeware projects, which will probably offer high-performance XML data storing in the near future. Examples for such projects are the XML databases XDBM (http://bowerbird.com.au/XDBM/), Lore (http://www.db.stanford.edu/lore/), and GMD IPSI’s XQL Engine (http://www.darmstadt.gmd.de/IPSI).

These projects are being closely observed and considered for further implementation.

The following figure shows the architecture of SIMPLEX:
SIMPLEX is written in Java 2. First, because it’s an open technology, and second, since there is a wide range of efficient XML components for java-based applications available. Free XML parsers and XSLT processors can easily be incorporated in a Java application. In order to be parser- and processor-independent we implemented the JAXP (Java API for XML Processing) technology, which decouples the application from the particular used parser and processor [8]. In our implementation, SIMPLEX uses the parser Xerces and the XSLT processor Xalan of Apache’s XML project (http://xml.apache.org).

For the GUI, we use the Swing and AWT classes available in Sun’s JDK and integrate the HTML ActiveX component of Microsoft’s Internet Explorer 5.x. This windows-specific component is wrapped into a Java class and is directly usable in a Microsoft Windows platform. By using tools like Linar’s J-Integra (http://www.linar.com/), we do not lose the platform-independent character of our solution, which is one of its characterizing principles.

The classes that are written particular for the HTTP-interface of Tamino realize the communication with the XML database. Every query and its result is transferred between SIMPLEX and the web server using HTTP. If the database is replaced, only these “Tamino-classes” either have to be changed or substituted. In our implementation, we use the open source web server Apache. For query statements, we apply XQL („XML Query Language“, see http://www.w3.org/TandS/QL/QL98/pp/xql.html). The result of the XQL queries and the data sent to the database is XML data. This data is parsed in SIMPLEX using Apache´s Xerces. This parser enables access to the XML-documents via the DOM and SAX interfaces.

Apart from the pure representation of XML-documents in different ways (see section 4.2), SIMPLEX offers two defined modules that take over important functions. The first module is the so-called RetrievalAgent. It supports the automated exchange of XML-documents without human intervention. The second module is STYX. This part of the application is responsible for the transformation of documents from one XML vocabulary into another (see section 4.4). These transformation capabilities are also used in SIMPLEX for the integration of XML-documents into in-house-systems. At this stage of the development, this integration is also possible via the XML database itself. Tamino is capable to integrate external data sources over the so-called X-Node using an ODBC interface (see section 4.5).

4.2. The structured representation of documents

The precondition for using SIMPLEX is the representation of the required business documents in an XML standard. A classification of business documents is already defined in SIMPLEX. These documents are Invoice, CatalogueQuery, DeliveryConfirm, DeliverySchedule, Forecast, InventoryQuery, MonitoringTransaction, OrderConfirm, Reclamation, SalesOrder and PurchaseOrder. The supported standardization initiatives are currently xCBL, OAGIS and eBIS-XML. We plan to support further vocabularies in the future.

The presentation possibilities offered by SIMPLEX include showing business documents as XML code, as a directory tree, as a table, and as a styled document. The tabular representation can be exported to Microsoft Excel for further analysis of the XML data. The following
figure shows the user interface of SIMPLEX. It presents the defined business documents of the supply chain on the left hand side. An XML purchase order in the XML code representation can be seen in the main part of the window.

4.3 The exchange of documents

Information exchange processes in supply chains can generally be divided into put and get processes. While in put processes the sender is actively delivering the information to the recipient, get means that the recipient has to fetch the desired information. SIMPLEX supports both put and get processes. The idea is represented in the following figure:

![Diagram](Filename)
In a put process of a supplier-customer-relationship, for example, the customer (buyer.com) will release an order (purchase order) in his in-house-vocabulary (OAGIS in figure 4). She or he has the opportunity to convert the document into the vocabulary (xCBL in figure 4) of the recipient (supplier.com) and store it over the Internet in the remote database.

In this case, employees of buyer.com will first log-in to their company’s database, then extract and transfer the desired order to the SIMPLEX application using XQL, check the order and send it to the supplier’s database. Access control is done by the supplier’s and the buyer’s web server.

The information exchange with SIMPLEX can also be automated. An example is a get process, in which the customer automatically fetches invoices addressed to him from the supplier’s database and transforms them without human intervention.

The module RetrievalAgent is responsible for this automated exchange. In the scenario shown in figure 4 the agent runs in the background of the customer’s system and periodically searches the supplier database for documents of the type Invoice. If the RetrievalAgent finds such a new document addressed to the customer, it will transfer the invoice into the own XML database.

4.4. The conversion of XML vocabularies

4.4.1. Conversion in SIMPLEX. The problem of the conversion between different data formats was already mentioned in the introduction. In the following, we assume that senders and recipients of business documents in the supply chain are all able to represent their data in XML. We also assume that these business partners use different XML formats, which best match their different needs for data holding. The STYX (STYX Translates Your XML) module was developed for converting XML vocabularies. The basic idea is that there is an available technology called XSLT, which allows the definition of style sheets for transforming from one XML vocabulary into another.

The XSLT syntax allows composing style sheets, in which every element of the initial document is associated with an element of the target document. The concrete addressing of the elements is implemented using XPath. The so called template-rules allow a content-specific conversion.

Below is an excerpt from a style sheet for converting an invoice from the xCBL 3.0 into the eBIS-XML format.

```xml
<PostCode>
  <xsl:value-of select="PostalCode"/>
</PostCode>
```

It shows how the `<PostalCode>` tag of the xCBL 3.0 document is converted into the corresponding eBIS-XML tag `<PostCode>`.

To carry out the document conversion, the user has to select the standard of the document to be converted and the standard the document is supposed to be converted into. STYX automatically looks up the suitable XSLT style sheet for the conversion and applies it to the source document. The STYX module already contains style sheets for the conversion between the vocabularies OAGIS, xCBL, and eBIS-XML. User-designed style sheets can also be included. Since information about used vocabularies and supported business document types is stored in a XML file, they can be edited dynamically at runtime.
The actual transformation is performed by the XSLT processor, which applies the style sheet to the source document and thus creates the desired target document. In the Internet a whole set of XSLT processors is available for free. The implementation in a language such as Java is simple; for instance, a transformation with Xalan needs in the most favorable case only three lines of code.

4.4.2 Conversion strategies. The need for an any-to-any-converter of XML-formats like STYX is not the most usual case. A participant of the supply chain will rather require a one-to-many converter for transforming incoming document formats into his in-house-standard and vice versa.

If there are \( n \) different partners and everyone uses a different XML-vocabulary that best fits his needs, this participant will require \( 2n \) style sheets to establish bi-directional conversion. The use of a style-sheet-based converter like STYX will then ensure compatible communication with his partners (see figure 6).

There is already a set of predefined style sheets in SIMPLEX, which can be customized and extended by the user. In addition, a graphical tool that is currently under development will support this customization process.

STYX is able to support both one-to-many and many-to-many-conversions. For this second case there are at least 3 possibilities, similar to network topologies that are worth considering (see figure 7).

It is hereby assumed, that bi-directional communication between the partners has to be supported:

(a) The standards are positioned in a ring structure; therefore, conversion takes place by passing the XML-document in a single, pre-defined direction along the ring, until the document finally is converted into the desired destination standard. This can only work properly, if there...
are no conversion losses between two standards. Due to the complexity and variety of available standards, this is unlikely to happen in practice. The advantages of this structure are on the one hand easy extensibility and on the other hand low development costs. If a new standard is introduced, only two new conversions have to be developed while one conversion is dismissed. Therefore, if \( n \) is the number of participants, there are always \( n \) conversion processes, which keep the development costs for the conversions relatively low.

A disadvantage of this structure is the low conversion speed, which comes from moving the document-to-be-converted along the ring. Therefore, with \( m \) intermediate standards, \( m+1 \) conversions have to be done. Furthermore, the ring structure is only applicable in situations, where all standard can be converted loss free into one another, which is due to the complexity and variety of standards rather doubtful in practice. When loss-free conversion cannot be fully ensured, eventual losses multiply when moving the document along the ring. Furthermore, if one style sheet is defective, many conversions cannot be done.

(b) For each possible conversion there exists a unique style sheet. Thus with \( n \) different formats a maximum of \( n(n-1) \) style sheets is needed. This has the disadvantage of being hard to extend. Contrary to (a), this structure has the advantage that occurring conversion losses are reduced to a minimum and do not alter the results of other combinations.

(c) The conversion between the standards is done by using an intermediate “super standard”. This standard \( S \) has to cover the full range of possibilities of all the involved standards to ensure a conversion with minimum loss of information. It is strongly questionable whether the requirements of a “super standard” can be met by any XML-vocabulary currently available. With \( n \) standards, \( 2n \) style sheets are necessary. The advantage of this structure is that additional standards can easily be supported by creating just two new style sheets.

![Figure 7. Strategies for a many-to-many-conversion](image)

4.5 The integration of the documents

The relatively simple processing of XML-documents facilitates their integration into in-house-systems, e.g. ERP-Systems, such as SAP R/3. The access to the XML-documents can easily be enabled using the DOM or SAX programming interface. The path to the SAP R/3 system goes through the BAPI interface or through the Business Connector.

Moreover, the integration with database systems is relatively easy [3]. By using Tamino, a relational database can quickly be integrated through mappings and an ODBC interface [13]. Some software vendors like Oracle...
also offer the applications that enable this integration (http://technet.oracle.com/tech/xml/). The new object relational concepts and appendages like “nested tables,” for example, facilitate the merger of the relational and the XML world [10][9]. Moreover, there are programming interfaces for most of the existing databases available, which allow the integration of XML into relational databases. For instance, the integration can be realized using DOM or SAX together with a JDBC driver (http://java.sun.com/products/jdbc/). In addition, we are currently developing a module for SIMPLEX, which will allow a graphical mapping of XML structures to database tables.

Another important aspect is the need of visualizing the business documents. One easy and fast way is transforming the XML-documents into HTML-documents by using an XSLT style sheet, in order to view them in a browser, for example. In SIMPLEX the XSLT processor Xalan runs this transformation, without changing the original data in any way. Hence, it is possible to create as many different visualizations of one document as the user of SIMPLEX needs. The following figure shows an example of the presentation of a purchase order to facilitate control activities of a user.

![Figure 8. Visualizing an XML-document using an XSLT style sheet](image)

### 5. Conclusion

In this paper, we present a prototype for supporting the document exchange in supply chains. The use of XML thereby plays a key role: The documents are described with XML syntax and transferred between the partners of the supply chain. The fundamental advantage is that the XML standard only defines a general language for the description of documents, yet it does not determine their content. That means that any kind of business document and thus, for instance, all currently available EDI standards can be represented with XML. Considering this, multiple standardization initiatives have meanwhile arisen, which define business vocabularies that are industry-specific or at least adapted to certain application needs. The partners in the supply chain can thus use XML as a common fundamental language building upon it different business vocabularies, which best meet their specific requirements.

As shown in this paper, the conversion between different business vocabularies using XSLT style sheets is possible. A translation between two standards is indeed still necessary; the advantage, however, is that this translation is relatively simple. Our experience shows that it is possible, with appropriate previous knowledge, to create such an XSLT style sheet in one day. Nevertheless, due to the different degrees of detail of the particular
vocabularies, a translation without loss of information cannot always be achieved.

Similar considerations, to the ones regarding the conversion, apply in the context of the integration into in-house-systems, since there is the same need of transforming an XML-document into another format. For instance, the elements of the document have to be mapped to the attributes of tables in a relational database system or to the slots of a business object. This process, however, is relatively simple to realize, due to the processing capabilities of XML.

At this stage of development, SIMPLEX just supports the exchange, the conversion and the integration of business documents based on XML. In a next step, we want to integrate the XML-data described in planning processes of the supply chain. The optimization of these extended planning processes is the fundamental thought of Supply Chain Management. The entire knowledge base of SIMPLEX will remain XML. This will make all the processing and structuring advantages of the XML family available to support inter-organizational planning and optimization procedures. Furthermore we want to test our solution in different live environments.

6. References


