Supporting B2B Business Documents in XMLWeb Services

KIM Hyoungdo
(Department of Management Information Systems, Hanyang Cyber Univ.  Seoul 133-791  Rep. of Korea)

Abstract While XML web services become recognized as a solution to business-to-business transactions, there are many problems that should be solved. For example, it is not easy to manipulate business documents of existing standards such as RosettaNet and UN/EDIFACT EDI, traditionally regarded as an important resource for managing B2B relationships. As a starting point for the complete implementation of B2B web services, this paper deals with how to support B2B business documents in XML web services. In the first phase, basic requirements for driving XML web services by business documents are introduced. As a solution, this paper presents how to express B2B business documents in WSDL, a core standard for XML web services. This kind of approach facilitates the reuse of existing business documents and enhances interoperability between implemented web services. Furthermore, it suggests how to link with other conceptual modeling frameworks such as ebXML/UMM, built on a rich heritage of electronic business experience.

Key words business document; XML web service; ebXML

These days XML web services become recognized as a business-to-business (B2B) solution as well as an integration framework for business applications. However, there are many drawbacks in XML web services to be useful for dynamic B2B applications as follows:

1) It should be possible to explicitly define and reuse existing business documents, traditionally regarded as an important resource for managing B2B relationships.

2) Security issues should be settled for guaranteeing secure B2B transactions in the open network environment.

3) More efforts should be made to standardize how to manage business transactions, how to express business processes, and so on.

Toward the complete implementation of B2B web services, this paper deals with the first issue of how to support B2B business documents in XML web services. Although XML web services support document-style interactions, they have functionally limited in explicitly managing business documents such as RosettaNet XML documents and traditional non-XML EDI documents. In most cases, designers must code first a web service with an API in order to derive it. While attractive for some applications, this kind of bottom-up approach deteriorates the interoperability and reusability of web services. As a solution, this paper presents how to express B2B business documents in WSDL [1], a core standard for XML web services. This kind of top-down approach facilitates the reuse of existing business documents and matches well with other conceptual modeling frameworks of service-oriented architecture (SOA) [2].

The rest of the paper is organized as follows. Section 2 reviews web services in the aspect of supporting B2B business documents. In Section 3, basic requirements for electronic processing of B2B business documents are summarized as a milestone for discussing alternatives in web services. Expressing B2B business documents in WSDL is then discussed in Section 4. Converting ebXML/UMM business transactions into web services is discussed in Section 5 [3,4]. Finally, section 6 concludes the paper with some future research directions.

Received 2004-06-19
1 XML Web Services and Business Documents

Web services support either RPC-encoded or document-style interactions. The former interacts with the back-end service in an RPC-like fashion. Typically, the interaction is a very simple request/response, where the client sends a SOAP message that contains a call to a method. The application server receiving this request can then translate this request into the back-end object.

With document-style interactions, XML business documents are transmitted across the businesses. They do not map directly to back-end method calls. They are actually complete, self-contained business documents. When the service receives the XML document, it might do some pre-processing on the document, perform some work, and construct the response back. There is usually no direct mapping to a back-end object. In fact, a request might invoke multiple components on the back end. Document-style messaging delivers a number of benefits. With document style, you can utilize the full capabilities of XML to describe and validate a business document. Document-style messaging does not require a tight contract between its service consumer and provider. Because documents are self-contained, document-style messaging is typically better suited for asynchronous processing.

WS Basic Profile 1.0 defines XML Schema, SOAP (Simple Object Access Protocol), WSDL (Web Services Description Language), and UDDI (Universal Description, Discovery, and Integration) as core specifications for web services interoperability[3]. WSDL exposes the methods of a software program and allows a remote user to invoke those methods by transmitting the variables and parameters of the operations in a SOAP envelope. SOAP messaging, independent of WSDL, enables XML documents to be routed and manipulated using the semantic content of the envelopes and/or documents. UDDI is the global look up base for locating the services defined using WSDL.

In a WSDL definition, a port type defines a web service, the operations that can be performed, and the messages that are involved in the operations. As an example, Fig.1 shows a WSDL definition together with an auxiliary diagram. The definition specifies that a port type ‘pt1’ has an operation ‘o1’ receiving message ‘m1’ and returning message ‘m2’. Related with business documents, message elements are important. Note that a message is composed of part elements that could be defined by XML Schema types and elements. Data types used in exchanging messages could be defined using the ‘types’ element within the root ‘definitions’ element.

Note that there is no explicit concept of business documents. Document types are not supported at all, meaning that message types are defined directly based on information items[6]. The notion of business document is at the heart of current business operations, and plays an increasing important role in B2B applications. Order, billing or delivery forms are all examples of transactional business documents that collect data which gives support to a certain business function, regardless of whether this data is finally stored in a database or in another remote repository. In order to support B2B business documents in web services, we need to define how to specify the requirements in WSDL. The requirements also invoke the introduction of a top-down approach to the development of web services, instead of traditional bottom-up approach in web services.

```
<definitions…>
  <message name="m1">
    <part name="part1" type="xsd:string" />
    <part name="part2" type="xsd:int" />
    <part name="part3" type="xsd:string" />
  </message>
  <message name="m2">
    <part name="image" type="xsd:binary" />
  </message>
  <portType name="pt1">
    <operation name="o1">
      <input message="tns:m1" />
      <output message="tns:m2" />
    </operation>
    …
  </portType>
…
```

Fig.1  A WSDL sample
2 Basic Requirements for Electronic Business Documents

Basic requirements for electronic processing of business documents are well defined in ebXML, built on a rich heritage of electronic business experience such as UN/EDIFACT. Here, the requirements are summarized as a milestone for discussing alternatives in web services.

In ebXML, a business transaction is an atomic unit of work in a trading arrangement between two business partners. As Fig.2 illustrates, a business transaction consists of a requesting business activity, a responding business activity, and one or two document flows between them. A business transaction may support one or more business signals that govern the use and meaning of acknowledgements. There is always a request document flow. A business transaction definition specifies whether a response document is required. This type of business transactions is typically associated with the formation of contracts or agreements. A business transaction with a request only is typically used for notifications.

Request and response document flows contain business documents that pertain to the business transaction request and response. Business documents have varying structures. Business signals, however, always have the same structure, defined once and for all. A document flow is not modeled directly. Rather it is modeled indirectly as a document envelope sent by one role and received by the other. The document envelope is always associated with one requesting activity or one responding business activity to specify the flow as the UML class diagram of Fig.3 depicts. Document envelopes are named. There is always only one named document envelope for a requesting activity. There may be zero, one, or many mutually exclusive, named document envelopes for a responding activity. In the actual execution of the purchase order transaction, however, only one of the defined possible responses will be sent. Each document envelope carries exactly one primary business document.

A document envelope can optionally have one or more attachments, all related to the primary business document. The document and its attachments in essence form one transaction in the payload in the ebXML message service message structure. Note that security requirements can be expressed for document envelopes and attachments respectively.

As an example, Fig.4 shows a document flow specification in ebXML.

3 Expressing B2B Business Documents in WSDL

In order to support B2B business documents in web services at the same level of business semantics as in ebXML, diversity of business documents and
attachments should be considered. If a document is already defined using XML Schema, it can be directly represented by a part element. There are two other situations that we should pay attention to. One is related with XML documents, not defined using XML Schema. For example, there are so many B2B business documents defined by DTD (Document Type Definition). If it is possible to develop a XML Schema description collateral with a given DTD, there is no problem. If it is impossible, we have to define the part for specifying the business document as anyType element. The other is how to cope with non-XML documents. For example, UN/EDIFACT EDI documents are still practically used in large businesses. As in the above case, it is a possible solution to define an anyType part for expressing each non-XML document. However, there is no way to express meta-data about the non-XML document, such as UN/EDIFACT EDI format. Furthermore, security and other requirements for business documents and attachments cannot be expressed in WSDL.

4 Top-Down Modeling Approach Based on ebXML

This paper focuses on how WSDL can be extended to specify the requirements. To this end, WSDL offers the documentation element that allows designers to provide further information about elements. It is possible to match elements between ebXML and web service such as DocumentEnvelope to message and BusinessDocument and Attachment to parts. The specific requirements for the messages and parts can be documented in WSDL as Fig.5 shows. In this case, the Order business document is an UN/EDIFACT EDI message and is attached with a jpeg image file. If a business document or an attachment is defined with a XML Schema, its type or element should be referenced by the type or element attribute of the part. Instead of this kind of expression, we may define a type definition composed of a business document and attachments. Parts in WSDL are a flexible mechanism for describing the logical abstract content of a message. A binding may reference the name of a part in order to specify binding-specific information about the part. So, defining a type definition for packaging a business document and attachments is not desirable.
sharing. This kind of bottom-up approach has many drawbacks. WSDL created from source code is less strongly typed than WSDL that is created from the original XML Schema. Creation of web services from source code implementations can lead to interoperability issues across platforms.

In many cases, it may be best to model services based on existing document formats. These may not be XML documents, but historically they do represent a sensible message or document within the context of a business process.

ebXML emphasizes business processes and defines semantically neutral core components, the reusable data items found in business documents. Standards-based business processes and document definitions will drastically reduce the cost of negotiating agreements. The ebXML business process modeling language and methodology of choice is the UML-based UN/CEFACT modeling methodology (UMM) and its supporting business process metamodel. The UMM metamodel is a mechanism that allows trading partners to capture the details for a specific business scenario using a consistent modeling methodology. The UMM metamodel supports a set of business process viewpoints that provide a set of semantics for each viewpoint and forms the basis of specification of the artifacts that are recommended to facilitate business processes and information integration and interoperability.

A hot issue is how to reuse the business process and information model in a technology-neutral manner. Business collaboration framework (BCF) is a new e-business framework using a technology and implementation neutral approach based on the UMM[7]. As a first step toward the direction, this paper discusses how to convert business transactions of ebXML into web services. WSDL has four transmission primitives that an endpoint can support:

1) One-way. The endpoint receives a message.
2) Notification. The endpoint sends a message.
3) Request-response. The endpoint receives a message, and sends a correlated message.
4) Solicit-response. The endpoint sends a message, and receives a correlated message.

An ebXML business transaction could be represented as a set of web services via various ways. Theoretically there could be 20 patterns of ebXML business transactions. Some patterns of ebXML business transactions directly match well with a transmission primitive. For example, ebXML business transactions of notification pattern can be directly converted into the notification primitive. However, most of business transactions correspond to a single primitive. It is necessary to define what combination of document flows and/or business signals corresponds to a web service. Basically, a document flow and its receipt acknowledgement signal could be combined to be a web service because the signal notifies that a message has been properly received. Acceptance acknowledgement signal needs to be treated as an independent web service because it signals that the message received has been accepted for business processing by the receiving application. So, a full ebXML business transaction is expressed by a combination of 4 web services. The combination of web services can be defined with a business process definition language such as BPEL4WS[8].

5 Concluding Remarks

Basic requirements and methods for driving XML web services by business documents are introduced in this paper. This kind of approach facilitates the reuse of existing business document standards and enhances interoperability between implemented web services. Furthermore, the top-down view of this approach makes it possible to link with other conceptual modeling frameworks such as ebXML/UMM.

Further research directions include the following:
1) Extending the capability of transformation from ebXML/UMM to web services. Business collaborations as well as business transactions need to be automatically transformed to a business process specification using a business process definition language.
2) Securing business documents in web services following the security requirements set in ebXML/UMM.

(Continued on page 73)