Extending Support for Contracts in ebXML

James Cole, Zoran Milosevic
CRC for Enterprise Distributed Systems Technology (DSTC)
The University of Queensland, QLD 4072 Australia
<colej, zoran>@dstc.edu.au

Abstract

This paper describes our proposal for extending the current ebXML standard - to provide more comprehensive support for electronic contracts. The paper first presents the current status of major B2B initiatives, focusing on their support for electronic contracts. Having found the ebXML meta-model to provide a suitable contractual foundation, we examine the full extent of the requirements for supporting electronic contracts, and propose extensions to the ebXML meta-model to facilitate these requirements.

1 Introduction

Realising the full potential of electronic business-to-business (B2B) exchanges requires comprehensive support for electronic contracts (e-contracts). To be effective, this support needs to be grounded in standards. The time is now ripe for providing such support. B2B commerce has really taken off in recent times, and there are numerous B2B standardisation efforts underway. Also, a major barrier to the use of e-contracts—lack of legal recognition of digital-signatures—has recently been overcome in certain jurisdictions, such as the US (see [1]), with others likely to follow in the near future—certainly by the standardisation is completed and systems based on the standards are in widespread use.

There is inadequate contract-specific support in existing computer-based B2B systems and standards. Though a number of them may make use of the term 'contracts', most tend to focus more on the practicalities of the exchanges and interactions between parties. While these are difficult problems in themselves, they do not address the specific semantic requirements of contracts, such as those for contract negotiation, and monitoring of the contract's conditions.

This paper presents an initial attempt at providing more comprehensive support for contracts in the ebXML meta-

model. Our work has been motivated by a desire to better support the semantics of contracts, in terms of what information electronic versions need to represent and what processes and issues surround them.

The structure of the paper is as follows. Section 2 summarises the current status of B2B initiatives with respect to the support they explicitly provide for e-contracts. Section 3 describes the support for contracts in the ebXML meta-model, which we find to provide the most comprehensive support of the current B2B initiatives. Section 4 identifies e-contracts requirements and then, based on these, proposes our extensions to the ebXML meta-model. Section 5 outlines our future work in this area and concludes the paper.

2 Current Status of B2B Initiatives

There has been a proliferation of various B2B initiatives over last couple of years. A significant enabler of this has been the development and widespread adoption of XML for the description of business documents, which followed rapid penetration of the Internet and distributed object technologies into businesses of various types. These B2B initiatives include:

- general consortia and standards bodies addressing a broad scope of electronic business interactions (e.g. ebXML [2], OBI [5] and BizTalk [6]).
- standards for certain vertical domains (e.g. RosettaNet [8] and I2I [15])
- research prototypes (e.g. COSMOS [9], BCA [22])
- specific vendor solutions (e.g. tpaML [11], WebMethods [12], mySAP.com [13], CommerceOne [14]),

This section summarises these initiatives with the aim of determining the extent to which they address e-contracts.

2.1 General B2B consortia and standards bodies

The ebXML (electronic business XML) [2] initiative is a standardisation effort established by the United Nations body for Trade Facilitation and Electronic Business

(UN/CEFACT) and the Organisation for the Advancement of Structured Information Standards (OASIS). ebXML's mission is to enable the "global use of electronic business information in an interoperable, secure and consistent manner by all parties" [3]. They have chosen XML as a basis for their standards. Although the initiative consists of a number of working groups, the Business Process group has accommodated support for contracts, as part of their business process sub-meta-model. One of ebXML's explicitly-stated general principles is "meeting businesses' legal needs" [1]—indicating a commitment towards the needs contracts serve.

OBI (Open Buying on the Internet) [5] is an open, vendor-neutral standard for B2B e-commerce. Currently, OBI focuses on procurement of high-volume, indirect products and services. It specifies a simple architecture based around a model of how a requisitioner from a buying organisation places an order via a catalogue at a selling organisation. In response to such an order the selling organisation sends back an order request, which can then either be rejected or approved. In the case of approval the buyer can send a completed order back to the seller. Another party that plays a part in this process is a Payment Authority. OBI is mainly concerned with the more practical details of electronic B2B interactions, such as the lower-level issues of interoperability, data formats, transport and security. They do not provide any direct support for contracts. The current OBI specification does not make use of XML as a data exchange format.

Microsoft's BizTalk [6] initiative is aimed at facilitating and integrating XML-based business processes within and between organizations for supporting e-commerce. BizTalk technology consists of:

- BizTalk Framework, which provides specifications for the XML-based messaging implementation – needed for transmission of business documents;
- BizTalk.org web site [7], which hosts a library of BizTalk XML schemas (representing commonly used business documents), BizTalk framework specification and a forum for developer community;
- BizTalk Server 2000 for server side document transformation and routing;
- BizTalk Jumpstart Kit (JSK) for client side document execution and business logic application. The components of the JSK are anticipated to be included as part of final BizTalk product release.

To the best of our knowledge there is no support for business contracts in BizTalk. The BizTalk Server's agreement tools provide a facility for routing XML messages between organizations, and in that respect there are similarities between BizTalk Server and the tpaML (described in section 2.4).

2.2 Vertical B2B standards

RosettaNet [8] is a standards initiative working on supply-chain standards. Their work focuses on four aspects of supply-chain interactions. They describe these areas using an analogy in human communication: at the lower-level are dictionaries, which are the basic elements, corresponding to words; on top of this is the framework layer, which corresponds to grammar; next is the partner-interface-processes (PIP) layer, which corresponds to a dialog between parties; and on top of that is their layer for Business Processes, named the eBusinessProcess layer. RosettaNet is focused largely on enabling processes, and support for contracts is not currently part of their standard.

BizTalk is also providing specialised XML-based specifications for domains such as government, health and finance.

2.3 Research Prototypes

The EU-funded COSMOS project [9] provides a set of services for facilitating the use of e-contracts. Rather than attempting to model the full complexity and semantics of contracts, their model identifies only those parts that they believe are amenable to efficient automation. Hence, much of the system deals with lower-level, communication and representation issues, though they do provide a basic architecture and a meta-model outlining the structure of a contract. In addition, they provide various tools, such as for contract-negotiation, and for assisting parties in automating the fulfilment of their contractual obligations, in the form of a tool for deriving a workflow (based on petri-nets) from a contract.

Our Business Contracts Architecture (BCA) [22] was one of the early research prototypes that proposed an architecture for business contracts. This architecture is described in terms of roles and their relationships which together support contract establishment, execution, monitoring and enforcement stages in a contract life cycle. Key roles in the architecture are as follows [10]:

- *Contract Repository*, to provide electronic repositories to store standard contract forms and optionally, standard contract clauses.
- *Notary*, to store signed instances of standard contracts forms, which can later be used as evidence of agreement in contract monitoring and enforcement activities.

- *Contract Monitor*, to enable monitoring of the business interactions governed by a contract and to signal the contract enforcer if violations are detected.
- *Contract Enforcer*, to enforce the compliance with contract terms. When signalled by the Contract Monitor, enforcer may send a warning notice to various parties informing them of the violation and possibly prevent further access to the system by non-conforming parties.

We note that there can be several business processes identified in this contract architecture, but our architecture is essentially role-based - to enable support for many types of underlying contracting scenarios (i.e. business processes implementing them).

2.4 Some major vendor solutions

The tpaML (Trading Partner Agreement Mark-up Language) is a recent IBM specification focused on specifying inter-organisational agreements, in terms of messages exchanged, message sequences and the underlying transport and security infrastructure [11]. These agreements are expressed using XML. To support agreement-related inter-party interactions, tpaML provides a number of tools and runtime services. The TPA tools include those to edit TPAs, register of the user application logic bound to the business's interfaces, and to facilitate code generation, essentially generating interfaces for each party's side of the business processes. The run-time services include the ability to monitor a TPAs' execution. The system is open enough that the specific interaction style and formats can be based upon standards such as OBI or RosettaNet. The tpaML specification was submitted to the ebXML consortium in January 2000.

WebMethods [12] provide a range of B2B software: WebMethods B2B, WebMethods B2B for Portals, and WebMethods B2B for Partners. These software packages enable B2B integration over the Internet and support most major standards such as RosettaNet, OBI, ebXML and BizTalk.

mySAP.com Marketplaces [13] is a web-portal providing a hub for many-to-many interactions between businesses. They provide a centralised system, hosting:

- infrastructure supporting the interactions, such as transaction integrity and security;
- applications (accessible using only a web-browser); directories for finding partners, goods and services; and
- services such as price negotiation and collaborative forecasting between multiple businesses.

These services are provided to remove as much of the infrastructure and administrative burden from the users as possible. Businesses can integrate with the portal in a simple fashion, via web-browser technology, or in a more integrated fashion if they are using compatible software such as the SAP R/3 ERP system. Though it is a proprietary solution, they plan to support emerging business access protocols and to publish their proprietary protocols. To the best of our knowledge, they do not provide explicit support for e-contracts.

CommerceOne MarketSite [14] is also a centralised many-to-many business hub. Thus, it provides the same benefits as explained for the similarly structured mySAP.com. MarketSite focuses on electronic procurement, and their system supports multiple e-commerce standards, such as OBI and RosettaNet—CommerceOne are themselves active in standardisation efforts such as RosettaNet. MarketSite does not provide as large a range of services as mySAP.com. The hosted applications they provide include a procurement application, and catalog/order management application, all of which can be accessed via a web-browser. They also host directories, such as trading directories. We are not aware of any support for contracts in the system.

i2i—industry-to-industry—is a web-based marketplace for buying and selling goods and services, specific to the chemicals, energy, retail and construction domains [15]. Using a web-browser, buyers and sellers can search, purchase and communicate. The site provides facilities for purchasing, which can be done via offers, trades and auctions. The site also offers directories and listings, and personalised views onto the information it contains. Contracting is not a service they provide, however; if parties decide to engage in business together contract arrangements must be made off-line, between themselves.

3 Contracts in the ebXML Meta-model

The existing ebXML meta-model [3] provides some basic support for contracts; this is situated in the *Resources and Contracts* grouping, shown in Fig. 1. The ebXML meta-model consists of five sub-sections, which they call 'groupings': Resources and Contracts, Markets and Parties, Business Processes and Rules, Business Service Interfaces and Communication, and Information Model.

The *Resources and Contracts* grouping represents a high-level economic model [3]. It has been adapted from the REA (Resources, Events, and Agents) ontology [16], which provides a minimal, and thus flexible and

adaptable, framework for describing economic exchanges. The main contract-related elements in the grouping are: Agreement, Contract, and Commitment; also found in the grouping are elements that provide an economic basis for contracts and the model as a whole: Economic Resource, Economic Event, and Duality. We discuss the elements of this economic basis first[1].

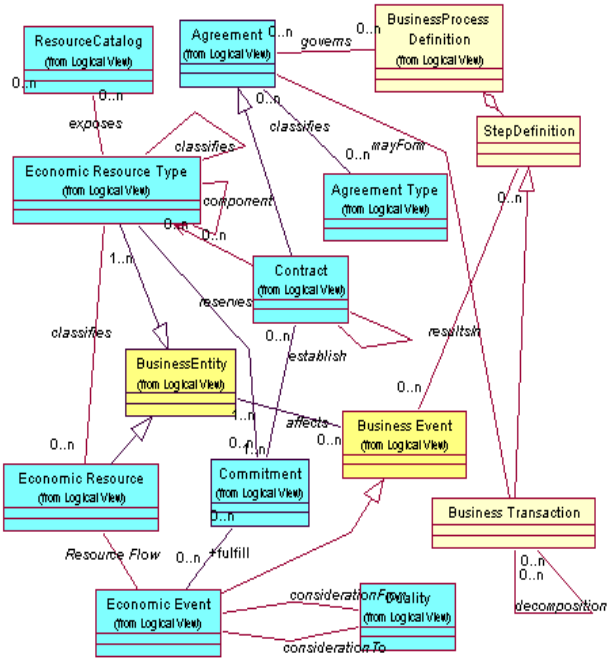

Figure 1: Resources and Contracts grouping in ebXML Meta-model

At the heart of the economic resources and exchanges is the *Economic Resource* class. *Economic Resources* represent "a quantity of something of value that is under the control of an enterprise. Examples are cash, inventory, labor service and machine service" [3]. An *Economic Event* represents the transfer of control of an *Economic Resource* between parties. A *Duality* is a "relationship between Economic Events, where one is the legal or economic consideration of the other", e.g. a payment in return for a product or service.

Turning our focus to contracts, an *Agreement* is an "arrangement between two parties that specifies in advance the conditions under which they will trade (terms of shipment, terms of payment, expectations of quotations and pricing, etc.)". In an *Agreement* there is no implication of specific Commitments for resource exchanges. A Contract is a specific type of Agreement for which there are such Commitments, that is "mutual arrangement[s] between parties that some actual economic exchanges will occur in the future". Commitments are thus an obligation to perform an exchange of an economic resource (i.e. to perform an action that results in an Economic Event).

Another grouping in the ebXML meta-model strongly related to contracts is the *Business Processes and Rules* grouping. This grouping is related to Agreements and Contracts in the following ways. An *Agreement governs* the of *Business Process Definition*s related to enacting it. A Business Process Definition consists of a number of *Step Definitions*, and the completion of one of these steps may *result in* the generation of an *Economic Event*; and this may in turn *fulfil* a *Commitment*. Basically, a Contract governs the processes that fulfil its commitments.

In summary, the ebXML meta-model provides a simple framework for contracts that has an economic basis.

4 Extending the ebXML Meta-model

In this section, we propose extensions to the ebXML meta-model to provide better support for e-contracts. We start by looking at the requirements for supporting e-contracts, and examine how the ebXML meta-model facilitates these requirements. We then present our meta-model, which attempts to better facilitate these requirements by extending the current provision for contracts in the ebXML meta-model.

4.1 Contract Requirements

There are a number of requirements for facilitating electronic B2B interactions. Firstly, widespread electronic B2B interactions require standards. There needs to be standards for: exposing a business' services, formats used for data being exchanged between businesses, standards for inter-business sequences of interactions, and so on. While these are difficult problems in their own right—currently receiving much attention through standardisation bodies such as OBI and RosettaNet to name a few—simply facilitating interactions between businesses is not the end of the story. Contracts are required to provide a legal basis for the exchange and thus reduce the risk of to the parties involved. For example, they ensure that the goods or services to be exchanged are well defined, and often provide formal means by which a party may terminate the agreement.

[1] Note that for brevity we only discuss, in the text, and only show, in the diagrams, the major elements and the major relationship between them.

In the following paragraphs, we look at e-contract requirements. Importantly, there needs to be support for legally enforceable contracts. In addition to this, there needs to be support for contract discovery and creation: discovering existing, standard contracts and contract elements associated with either a process, company or domain and, perhaps using these as a staring point, creating and negotiating contracts, with the end goal of having a signed contract instance. Another requirement is for support and automation in the deployment of contracts: in creating computerised systems to help parties meet their contractual obligations, by automating the necessary business processes, providing contract-related information to the parties, and, in some situations, by monitoring for contract violation and perhaps attempting to enforce the contract's conditions.

Providing legal enforceability is an important requirement for e-contracts. In [17] four requirements for a contract to be legally valid are given: legal purpose, capacity or competence of the parties involved, clarity and consideration. A contract has Legal Purpose if what it entails is legal within the jurisdiction that governs it. The parties involved in a contract have Capacity or Competence if they "are lawfully capable of agreeing to contracts (e.g. whether an individual has the authority to represent their organisation)" [3]. This can be checked using a means described in [18]. A contract has Clarity if it is unambiguous. A contract has Consideration if it involves exchange(s) between the parties involved. In addition to these requirements, a legally enforceable contract must also contain the parties' signatures: digital signatures can be used for this purpose in a jurisdiction where they are legally recognised. And finally, these signed contract instances need to be stored in a safe repository to prevent tampering or accidental loss.

Contracts may be written from scratch, but often are based upon standard templates and clauses—standard within a company or domain; contract systems should support this use of standard items. It follows that should be repositories for storing these standard items, and means of finding and navigating these repositories. Supporting the use of standard items should generally help to reduce the burden of ensuring contracts are legally-valid, as the legality of the standard items would presumably be known and have been tested in practice. Another requirement is being able to discover contract(s) associated with a service provided by a party. Finally, negotiation is a important part of creating a contract; to facilitate it, support is required for the back and fourth dialog of proposals and counter-proposals, and the changes in the contract document they entail.

Deployment comes after a contract has been signed by the parties, and includes the tasks they undertake in setting up their systems to support their contractual obligations, as well as their processes and interactions in attempting to meet these obligations. Deployment involves:

- developing systems to support the parties in enacting the behaviour required of them by the contract;
- monitoring of the parties' behaviour for compliance to the contact;
- providing information to the parties, such as notifications of when a contract is about to expire; and
- reaction to a violation of the contract, which may result in attempts to enforce the terms of the contract. For example, automatically incurring a fine on the supplier every hour after the due date for the arrival of undelivered goods.

In the general case, a large proportion of contract-deployment tasks are not automatable, either from a current lack of adequate expressive means or infrastructure, or for more inherent reasons. For example, a computerised system may not have adequate input to perform monitoring, such as for a clause in a land tenure application contract requiring that a 'for sale' sign be placed in a publicly visible position for the duration of the tenure. However, some contract-deployment tasks are automatable, such as monitoring that a payment has been made or providing notifications of significant dates in the contract, to name a few.

In the following paragraphs, we outline how the ebXML meta-model supports these contract requirements—legality, discovery and creation, and deployment.

In terms of legality, the ebXML meta-model provides no construct to represent a legally valid contract—the nearest candidate, the Contract element, contains only the commitments a party has for providing goods and/or services. Nor are there direct ways to represent that: a contract has legal purpose within a jurisdiction, the parties involved have the required competence property, or that a contract has the clarity property. The Duality element could, however, be used to show that the contract includes the requisite consideration. While the ebXML standard does include the notion of digital signatures, they are not currently included in the meta-model.

In terms of the discovery and creation requirements, there is no direct support for standard contracts, standard clauses, a contract repository or negotiation. However, there is the notion that a Business Transaction may result in (*mayForm*) the creation of an Agreement, and there is an *Agreement Type* class, which is used to classify different types of agreements.

In terms of deployment and providing support for a party enacting its contractual obligations, there is no means to associate a contractual condition with a business process for fulfilling it. Such an association would enable having "off-the-shelf", standard business processes for commonplace commitments. In terms of contract monitoring, only a small amount of the information found in a real contract can be specified directly using the meta-model, only the Commitments involved—the exchanges the parties agreed to—and thus only basic information relating to these could be monitored. But even here, there exists no means to associate a Commitment with the business process(s) and/or role(s) that could perform its monitoring. Similarly, adequate expressive means are lacking for associating with the contract, processes and roles for reacting to contract violation, or for associating elements that specify informational notifications relating to a contract to the parties.

While the ebXML meta-model does not adequately support all the requirements, we believe that it provides a solid framework which can be augmented to provide more comprehensive e-contract support.

4.2 Our Meta-model

Our extensions to the ebXML meta-model are intended to better facilitate the e-contract requirements identified

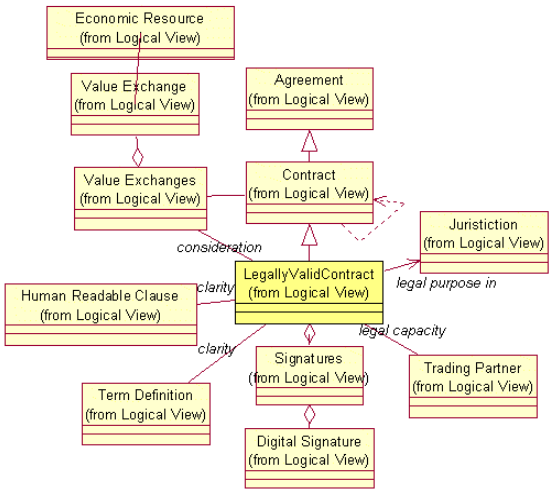

Figure 2: Legality package

above. Our meta-model represents an initial attempt at meeting these requirements; it is not a complete and fully worked-out meta-model and requires further work to improve its precision and formality.

We begin by giving an overview of our definition of the "internal" structure of a contract, in order to facilitate the exposition of our extensions. In the ebXML meta-model, the internal structure of a contract is left open (i.e. undefined), but we have found it useful to specify certain aspects of their structure, to facilitate achieving the above-stated requirements.

In our meta-model, contracts consist of a number of Clauses (where 'Clause' corresponds to the standard usage of the term: a distinct condition of the contract). Associated with each Clause may be:

- the human-readable version of it, which is equivalent to a clause in a traditional contract. (the other elements described below are used to assist in the computerised handling of the contract);
- roles, to be filled by the parties involved in the contract;
- economic resources—individual clauses may reference economic resources, in detailing a specific aspect of their exchange;
- term definitions, which precisely define the meaning of terms used in the clause; and
- policy statements, which describe the clause in a machine-readable form.

In addition to the Clauses it contains, a Contract has associated with it:

- roles and the trading partners that fill them. For example, the 'supplier' and 'customer' roles, and the companies that fill each of them;
- value exchanges, which are the resources that will be exchanged under the contract's conditions.

Legally Valid Contract overrides some of the Contract element's associations, and adds a few of its own. These associations specify the:

- jurisdiction(s) within which the contract has legal purpose;
- trading partners that have legal capacity, via the *capacity* association;
- value exchanges, which form the specification of the contract's legal consideration (via the *consideration* association);
- clause and term definitions, via the *clarity* association, indicating that they have the required legal property of clarity; and the
- digital signatures of the trading partners.

Having show how we describe contracts, we now move onto a description of our attempt to meet the requirements for e-contracts in our meta-model, through this richer contract representation and other additions.

4.2.1 Legal Validity

The Legal package in our meta-model is shown in figure 2. In it, a legally valid contract is represented by the Legally Valid Contract element, which we have added as a sub-type of the Contract element. This class is governed by a number of constraints. There is a mandatory[2] *legal purpose in* association between it and a Jurisdiction element (which we have added to represent a legal jurisdiction). To represent the information required by the competence requirement, we add a mandatory association named *competence* between Legally Valid Contract and Trading Partner. Consideration is represented via a mandatory association between Legally Valid Contract and the Value Exchanges class; each Value Exchange consists of two complementary exchanges. We can represent the information that a term-definition, clause or entire contract has been *checked* for clarity, by introducing a Boolean *clarity* property for each of them; thus a constraint can be added to the Legally Valid Contract element to ensure that all its constituent items have a *true* value for this property. The parties' signatures are represented by a mandatory association from Legally Valid Contract to the Signatures element, which aggregates the individual digital-signatures (Digital Signature elements). There must be a Signature for each party in the contract. The Signatures class is linked via a mandatory association to Legally Valid Contract. We have also added a Notary element, which is a role filled by the entity that stores signed contract instances. It has been added as a subtype of the ebXML meta-model's Role class.

4.2.2 Discovery and Creation

The Discovery and Creation package in our meta-model is shown in figure 3. For the discovery of contracts that are associated with a particular company, we have added a connection between a Business Service (an element in the *Markets and Parties* grouping in the ebXML meta-model, used to represent services offered by a party) and the Contract that governs it. Our Clause element provides a means for creating contracts from standard elements (standard clauses). Discovering a domain's standard contracts and clauses is facilitated by associating its Market (from Markets and Parties) with a Repository (which we have added as a sub-type of the Role class). A repository aggregates (standard) Contracts and Clauses. While a meta-model is not the place to provide detailed support for negotiation, we have created a *Negotiation* sub-type of the *Business Process Definition* element for identifying processes used for negotiation. In this vein, we have added to our meta-model processes for validation (Validation) and signing (Signing) and an associated Validator role (one is not required for signing). Note that our Negotiation element has a *results in* association to Legally Valid Contract, which replaces the *mayForm* association between a Business Transaction and the Agreement in the ebXML meta-model.

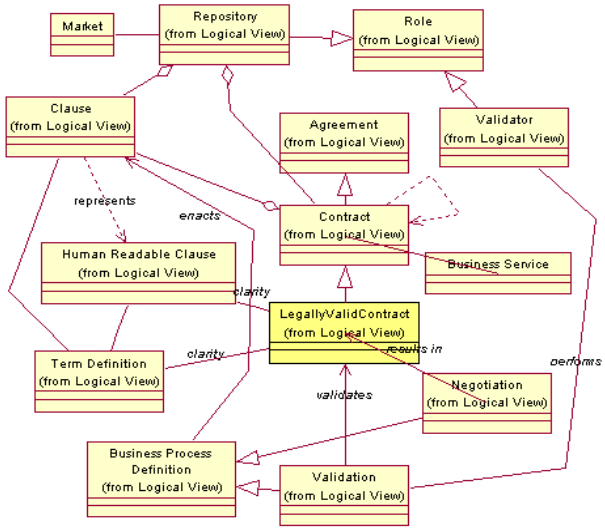

Figure 3: Discovery and Creation package

4.2.3 Deployment

Our deployment package is shown in figure 4. We facilitate automation in parties enacting their contractual obligations by adding an *enacts* association between Business Process Definition and Clause. This association is intended to facilitate the use of standard clauses and contracts; standard contract elements can have associated standard business processes for their enactment. For providing parties with informational messages related to a contract, such as a notification of a payment due date, we have added a *Notifier* class as a sub-type of Role, and associated this with Contract. How the events to provide notifications for and what information to provide in the notification would be specified are beyond the scope of this paper. For the monitoring and reaction to contract violation, we have introduced the notion of *policies* into the meta-model.

Policies are declarative rules that specify constrains on the possible behaviour of agents within a system: the semantics of a contract can be seen as a collection of a policy statements. The purpose of a contract is to constrain the possible behaviour of the parties to ensure a fair exchange. The policies field is young and there are

[2] We use the term *mandatory association* to mean an association that all instances of the particular classes must have.

many types of policies that only human can understand; however computers can understand simple policies (e.g. for access control) and their capabilities to understand, monitor and attempt to enforce policies is sure to grow. As presented in [19] and [20], we have done work on the specification of policies, including representing them in XML.

Our view of policies is based upon that of the ODP (Open Distributed Processing) Enterprise Language specification [21]. In ODP there are three basic types: obligations, permissions and prohibitions. We have introduced elements representing each of these three policy types. Obligations state that particular behaviour is required. Permission denotes that a particular behaviour is allowed: managers are allowed to hire staff. Prohibitions are typically the flip side of permissions: they denote

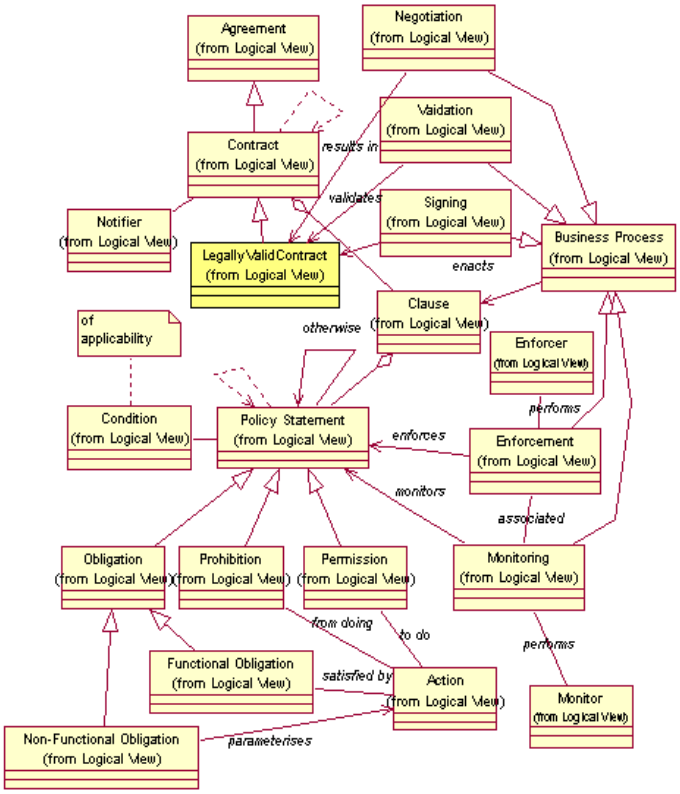

Figure 4: Deployment and Policies package

something that a party is *not* allowed to do: employees are not allowed to take holidays in their first three months of employment. We have further subdivided obligation into two types: functional obligations, which specify a specific action that must be undertaken, such as "clerks must balanced accounts every Friday", and non-functional obligations, which specify how actions must be undertaken, such as "all transactions made by the parties in a contract must use the specified currency".

Policies are associated with a number of elements. A Clause may be associated with Policy statement(s)—representing a computer-readable version of it. Via this association Policies are indirectly associated with the parties they constrain the possible behaviour of. They are also associated with the Actions they constrain the performance of. For non-functional obligations, this association is named *parameterises*—non-functional obligations specify (i.e. parameterises) *how* the action should be performed. For functional obligations, the association with Action is named *satisfied by*—performing the action satisfies the obligation. A Policy element is associated with a Condition element, which represents a policy's condition of applicability—when it should be in force. For example, a worker may be able to approve an order, *if* it is below $200. Finally, a Policy Statement may have a recursive *otherwise* association specifying a policy statement that should come into force if it is violated. They also have a recursive *dependency* association, specifying dependencies on other policy statements.

For monitoring contract conditions, we have introduced a Monitor element (as a sub-type of Role) and a Monitoring element (sub-type of Business Process Definition). There is an association between Monitoring and Policy. For example, an entity filling a Monitor role could be parameterised with the appropriate policy statements, and detect violations such as a payment becoming overdue—it could monitor the message exchanges for the payment to occur. The monitor might be hosted on a trusted, external server. The monitor could also note when the contract conditions are satisfied, such as the transferral of a payment. Each of the parties could also have their own internal monitors that serve in a guidance-type role, and which might help them avoid accidental contract violations. Similarly to the Monitor and Monitoring classes, we have introduced Enforcer and Enforcement classes—implementations of these would attempt to enforce the contract conditions. As not all clauses can necessarily be mapped to policy statements, we have added an association between Clause and both the Enforcement and Enforcer elements, which would be filled by manually written entities in an implementation.

Note that there is a distinction between contract templates (a contract without its particulars, such as 'start date', filled in) and contract instances (that are signed and have their particulars filled in), but that due to space restrictions we have omitted this distinction from the meta-model and its subsequent discussion.

5 Conclusion and Future Work

In this paper we presented an extension to the ebXML meta-model for providing more complete e-contract support. The ebXML meta-model was found to be the best starting point for providing standards-based support for contracts, after examining some of the major B2B initiatives and finding them to be generally lacking in their support for e-contracts. We have formulated a set of requirements for supporting e-contracts, and this was used as a basis for our extensions.

In our extensions, we have attempted to cover the types of contracts we are aware of but we acknowledge that the concepts we define may not be useful or adequate for all types of contracts. However, our extensions should not exclude other forms of contracts: our extensions don't have to be used as a complete whole, and the solid, general base provided by the ebXML meta-model should allow different types of contracts to be modelled, for which it may be useful to draw upon the suitable parts of our extensions.

This work is a presentation of our initial ideas on this subject, and there are many issues open for refinement and further elaboration. In particular, there are questions relating to the requirements for e-contracts. For example, what details related to a contract need to be represented and what is the best way to structure a contract, in order to facilitate their use, re-use, etc. To improve our understanding of the requirements for e-contracts, we plan to examine them from a more practical standpoint. In particular we wish to focus on the contracts, associated documents and associated process in a particular domain—the financial domain—and as part of this, examine the contract templates being developed within the Australian Superannuation Special Interest Group [23]. The experience we gain from this can feed back into the design of our Meta-model. Another area for future work is positioning this work with the Business Process model in our EDOC (Enterprise Distributed Object Computing) profile for UML [24].

Acknowledgements

The authors would like to thank Kerry Raymond for her comments on an earlier version of this paper.

The work reported in this paper has been funded in part by the Co-operative Research Centre Program through the Department of Industry, Science & Tourism, Australia.

References

[1] Costello, S. Clinton signs digital signature bill, IDG News Service www.civic.com/civic/articles/2000/0703/web-1esign-07-03-00.asp, 7 March 2000.
[2] ebXML. www.ebxml.org
[3] Business Process Project Team, The ebXML Business Process Meta-model Second Draft 6/23/00, available from www.ebxml.org/specdrafts/Busv2-0.pdf
[4] www.ebxml.org/specdrafts/TechSpechv0-6.pdf
[5] OBI: www.openbuy.org
[6] BizTalk Initiative: www.microsoft.com/biztalk/
[7] BizTalk Org: www.biztalk.org
[8] RosettaNet: www.rosettanet.org
[9] Griffel, F., et al. "Electronic Contracting with COSMOS – How to Establish, Negotiate, and Execute Electronic Contracts on the Internet" EDOC'98 Workshop, La Jolla, California USA, November 1998
[10] Goodchild, A., Herring, C., Milosevic., Z., "Business Contracts for B2B", CAISE'00 Workshop on Infrastructures for Dynamic B2B Service Outsouring, Stockholm, June 2000.
[11] tpaML: www-4.ibm.com/software/developer/library/tpaml.html
[12] WebMethods: www.webmethods.com
[13] mySAP.com Marketplaces: www.mysap.com
[14]: CommerceOne MarketSite: www.commerceone.com
[15]: i2i: industry-to-industry: www.i2i.com
[16] William E. McCarthy – REA papers: www.msu.edu/user/mccarth4/
[17] Reinecke et al., *Introduction to Business – A Contemporary View*, Allyn and Bacon, 1989.
[18] Milosevic, Z., Arnold, D., O'Connor, L. "Inter-enterprise Contract Architecture for Open Distributed Systems: Security Requirements". WET ICE'96 Workshop on Enterprise Security, Stanford, USA, June 1996.
[19] Herring, C. and Milosevic, Z. "Implementing B2B Contract Using BizTalk", HICCS'34.
[20] Cole, J., Derrick, J., Milosevic, Z., Raymond, K., "Author Obliged to Submit Paper before 4 July: Policies in an Enterprise Specification", submitted to Policy2001 workshop, Bristol, UK, Jan 2001.
[21] ISO/IEC JTC1/SC21, "Basic reference model of open distributed processing, part 2: Descriptive model," ITU-T X.902 – ISO/IEC 10746-2, Aug. 1994.
[22] Milosevic, Z. and Bond, A. "Electronic Commerce on the Internet: What is Still Missing?"; Proceedings of 5th Conference of the Internet Society; pages 245-254, Honolulu, June 1995.
[23] "Superannuation Industry EC Standards" http://www.superec.org/
[24] DSTC "DSTC Initial UML Profile for EDOC RFP submission" http://cgi.omg.org/cgi-bin/doc?ad/99-10-07