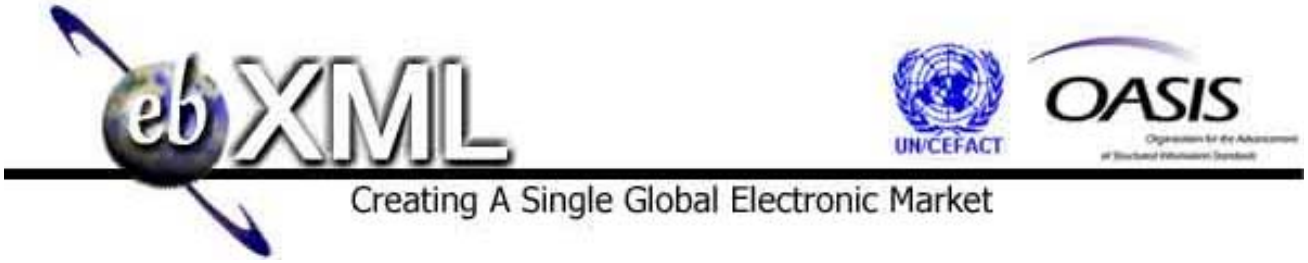

E-Commerce Patterns

v1.0

Business Process Team

11 May 2001

(This document is the non-normative version formatted for printing, July 2001)

Table of Contents

1 Status of this Document

This document specifies an ebXML Technical Report for the eBusiness community.

Distribution of this document is unlimited.

The document formatting is based on the Internet Society's Standard RFC format.

This version:

www.ebXML.org/specs/bpPATT.pdf

Latest version:

www.ebXML.org/specs/bpPATT.pdf

2 ebXML Participants

Business Process Project Team Co-Leads:

Paul Levine	Telcordia
Marcia McLure	McLure-Moynihan, Inc.

Business Process/Core Components Joint Delivery Analysis Team Lead:

Brian Hayes	Commerce One

We would like to recognize the following for their significant contributions to the development of this document.

Editor:

Jamie Clark,	McLure-Moynihan, Inc.

Contributors:

Bob Haugen	Logistical Software LLC
Nita Sharma	Iona
David Welsh	Nordstrom.com

3 Introduction

3.1 Summary

This document is a supporting document to the ebXML Business Process Specification Schema [ebBPSS], to address common pattern implementation issues and provide examples. The 'Simple Contract Formation Pattern' defined here demonstrates a non-normative rule-defined subset of BPSS use for practical contracting purposes. It also is aligned with the "drop ship vendor" model collaboration used by the Worksheets published by the ebXML BP/CC Analysis Team. The 'Simple Negotiation Pattern' defined here demonstrates a non-normative rule-defined subset of BPSS use to allow simple exchanges of 'dry run' transactions and collaborations that may result in a collective decision by trading patterns to use them on an enforceable basis. It also may be suitable to automate the negotiation of ebXML CPA terms from CPPs.

3.2 Audience

This document is intended to be read by designers and implementer of ebXML business processes.

3.3 Related documents

[ebTA] ebXML Technical Architecture Specification, Version 1.0.4, 16 February 2001. ebXML Technical Architecture Project Team.

[ebBPSS] ebXML Business Process Specification Schema, Version 1.01, 11 May 2001. ebXML Context/Metamodel Group of the Business Process/Core Components Joint Delivery Team.

[ebGLOSS] ebXML TA Glossary, 11 May 2001. ebXML Technical Architecture Team.

[ebCPP] ebXML Collaboration Protocol Profile and Agreement Specification, Version 1.0, 11 May 2001. ebXML Trading Partners Team.

UN/CEFACT Modelling Methodology, Version 9.1. 2001. UN Economic Commission for Europe. (CEFACT/TMWG/N090R9.1) [UMM]

Commercial Use of Electronic Data Interchange: A Report and Model Training Partner Agreement. 1992. American Bar Association Section of Business Law. [http://www.abanet.org/buslaw/catalog/5070258.html] [ABA Model Trading Partner Agreement 1992]

The Commercial Use Of Interchange Agreements For Electronic Data Interchange, UN/ECE Recommendation No.26. 1995. UN Economic Commission for Europe. (TRADE/WP.4/R.1133/Rev.1) [http://www.unece.org/trade/untdid/texts/d240_d.htm]] [UN/ECE Interchange Agreements for EDI 1995]

3.4 Document conventions

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in RFC 2119 [Bra97].

4 Design Objectives

4.1 Problem description

The BP Specification Schema [ebBPSS] contemplates exchanges of *Business Documents* composed into atomic *Business Transactions* each between two parties. In order to achieve the desired legal and economic effects of these exchanges, the structure of the *Business Transactions* must

- generate a computable success or failure state for each transaction that can be derived solely from the application of the ebBPSS standard and the data exchanged in the Business Documents and Business Envelopes,
- permit the parties to exchange *legally binding* statements and terms,
- permit the parties to exchange *nonbinding* statements and terms, in order to negotiate, and
- permit a logical composition of those exchanges into *Collaboration* patterns that allow agreements about sequences of transactions to be formed.

4.2 Terminology

4.3 Significant terms defined in ebXML

Business Collaboration -- The "Business Collaboration" object as defined in [ebBPSS].

Business Document -- The "Business Document" object as defined in [ebBPSS].

Business Transaction -- The "Business Transaction" object as defined in [ebBPSS].

Contract – Generally, a bounded set of statements and/or commitments between trading partners that are intended to be legally enforceable as between those parties. [ebGLOSS]

Legally Binding – An optional character of a statement or commitment exchanged between trading partners (such as an *offer* or *acceptance*), set by its sender, which indicates that the sender has expressed its intent to make the statement or commitment legally enforceable. [ebGLOSS]

4.4 Terms defined for the purpose of this document.

Acceptance -- A responding party's *document* indicating agreement with a received *offer*.

Binding -- See "Legally Binding" above.

Business Signal Parameters -- The following parameters as defined in [ebBPSS]:

isAuthorizationRequired	timeToPerform
isIntelligibleCheckRequired	isAuthenticated
isNonRepudiationRequired	isConfidential
isNonRepudiationOfReceiptRequired	isLegallyBinding
timeToAcknowledgeReceipt	isTamperProof
timeToAcknowledgeAcceptance	isGuaranteedDeliveryRequired

Collaboration -- See "Business Collaboration" above.

Counteroffer advice -- A message bound *to a rejection*, indicating that the sender intends to send a new *offer* regarding the same subject matter.

Document -- See "Business Document" above.

Offer -- A *document* proposing business terms by a requesting party addressed to a responding recipient. A *binding* offer entitles the recipient to form a *contract* with the requesting party by responding with a *binding acceptance*.

Nonbinding -- An optional character of a statement or commitment exchanged between trading partners (such as an *offer or acceptance*), set by its sender, that indicates the intent to be *legally bound*. See "Legally Binding" above.

Rejection -- A responding party's *document* indicating that it rejects a received *offer*.

Transaction -- See "Business Transaction" above.

4.5 Assumptions and constraints

4.6 Constraints from legal and auditing requirements

a.) **Enforceability requires an expression of intent.** In order for a message to be given legally enforceable effect, whatever its form, the author must indicate his intent to be bound. The message's sender may accomplish this by intentional use of a standard that specifies a mark, attribute or protocol indicating legal assent. In a paper context, this might mean affixing a written signature, plus an absence of elements that qualify its enforceability. (Elements that might tend to do so could include a substantive precondition to enforceability, the omission of essential terms, or a 'draft' stamp on its face that impeaches the document's finality).

b.) **Each offer must succeed or fail.** The *offer* in a *binary transaction* must be definitively resolved in order to end the *transaction*. (This is true whether or not the offers are *binding*.) Offers that are followed by an explicit *acceptance* must be resolved as accepted. All other responses – including time-outs, rejections and counteroffers – must be resolved as a type of *rejection*. Either resolution should result in completion of the *transaction*, together with a suitably provable "success" or "failure" end state that informs further processing of the results of the *transaction*.

c.) **Each acceptance must relate precisely to an offer.** Each *acceptance* of an *offer* (whether or not *binding*) must unambiguously refer to the *offer* accepted, in a manner that produces artifacts transmitted between the parties and suitable for proving the identity of the terms that were *accepted*.

d.) **Replicable and computable transaction state closure.** In the foregoing context, "suitable proof" of the offer and acceptance events, means that determinable computation of the *transaction's* "success" or "failure" state must be replicable by both trading partners at run time, as well as third parties (such as a court) after the fact, using only artifacts transmitted within messages associated with the *transaction*.

A sidebar: Nonrepudiation and Enforceability

Users of this document should note that the *defined signals* isNonrepudiation-Required, isNonRepudiationOfReceiptRequired and isLegallyBinding are significantly distinct from the *generalized goals* of nonrepudiation and legal enforceability. Invoking the former should assist, but does not assure, the latter. The goal of a well-designed electronic commerce model is to *reduce* the risk of repudiation and unenforceability to a reasonable minimum. *No* system will completely eliminate either risk. See [ABA Model Trading Partner Agreement 1992] and [UN/ECE Interchange Agreements for EDI 1995].

Repudiation risk occurs *whenever* a trading partner has an opportunity to avoid the consequences of its commitments. For example, under the BPSS, if you impose an timeToAcknowledgeAcceptance parameter (time>0) on a trading partner's response to you, he may validly reply with an exception claiming that your requesting document does not conform to the relevant business rules. That claim may or may not be true: in fact, nothing in the standard *computationally* prevents him from making a false exception at runtime. That opportunity may be the functional equivalent for him of a chance to repudiate. Say your requesting document offers to buy 1000 units of X. Assume you and he have a pre-existing contract requiring him to sell you 1000 units of X whenever you offer to buy them. He may have received, parsed and understood your requesting document as a purchase order to buy X. But he is still in a position to inaccurately claim that your purchase order failed a business rule check. Perhaps he has a limited supply of X, and a buyer who will pay more than you. *At run time*, there likely is no way for you to tell.

What *business signal parameters* offer, in that instance, is a set of process rules that require you or him to keep and store significant artifacts from the transactional messaging, that *later* may be impartially interpreted. Any "legally binding" obligation should, as a design matter, generate a set of those artifacts that would be useful in proving later in court that (for example) the claim of a failed business rule check was fraudulent.

In the electronic commerce context, an evaluative judgment that a set of messages creates an *enforceable* or *nonrepudiatable* contract should be understood to mean that the quality and coherence of the evidentiary artifacts available to prove it are acceptably strong. We *cannot* prevent trading partners from lying. We *can* design signal structures that make it easier to prove later.

4.7 Constraints from ebXML structure and standards

a.) **Business Service Interface.** An ebXML *collaboration* is conducted by two or more parties, each using a human or an automated business service interface that interprets the *documents* and *document envelopes* transmitted and decides how to (or whether to) respond.

b.) **Decomposition of business processes into binary pairs.** All *collaborations* are composed of one or more atomic *transactions*, each between two parties. Multi-party or multi-path economic arrangements are possible, and may be arranged in a single *collaboration*, but must

be decomposed into bilateral *transactions* in order to be modeled and executed under the ebBPSS.

c.) **Definitive use of visible end state machines.** The ebBPSS uses guard expressions that permit the reliable computation of *transaction* "success" or "failure" transaction end states. For the sake of reliability, these must be the exclusive source of instructions to the trading partner's business service interface, within the scope of that *transaction*. Any contingency or business logic that is to govern the reaction of the business service interface to a *transaction* must be expressed within the relevant *collaboration* in a manner that affects the end state, and that manner must be made visible to both trading partners in the business process specification referenced by the CPA to which the partners agreed.

d.) **Function of digital signatures.** Several ebXML specifications permit electronic signatures (generally conforming to the W3C XML-DSIG standard) to be used for various purposes such as message integrity or sender identification. Therefore, the presence or absence of an electronic signature bound to a *document* by hashing or the like, cannot, by itself, be used to indicate the *document's binding* character.

e.) **Ability to declare documents nonbinding.** The ebBPSS permits a trading partner to explicitly designate specific *documents* as *binding* or *nonbinding* by setting the Boolean parameter "isLegallyBinding".

5 Contract Formation in ebXML

5.1 ebBPSS contract formation functionality

The constraints listed in Section 4 provide implementers with a specific set of tools for producing reliable artifacts to evidence contracts. The ebBPSS constrains process designers and implementers to two methods of affecting the determination of a transaction's "success" or "failure" end states:

1. The semantic contents of the *documents* and *document envelopes* that pass between the trading partners can be referenced and evaluated in a guard expression, and

2. The BPSS *business signal parameters* that resolve requests for acknowledgement and the like, short of substantive responses to BusinessDocuments.

In the context of simple contract formation, trading partners may explicitly form a contract by exchanging requesting documents constituting *binding offers,* and responding documents constituting *binding acceptances,* resulting in a demonstrably successful or failed negotiation of the business terms proposed in the offer.

A sidebar: Explicit vs. implicit contracts

There is an important distinction between the legal view of contracts and this document's definition of "contract". The former encompasses a much broader range of phenomena that may be interpreted as a enforceable agreement.

In commerce, some agreements are formed by reciprocal actions and implied promises, without any explicit messages in one or both directions. If one trading partner acts in a manner that reasonably seems to convey an offer to sell an object, and the other partner carts off the object, a court may conclude that the latter's behavior is acceptance by performance. In such a case, the *implicit contract* is formed by *inferring* acceptance, *as if* the latter party had explicitly accepted an explicit sale offer.

In this document we are only concerned with exchanges of explicit messages that, if they logically match, will produce an explicit contract expressed in and evidenced by the messages. However, process designers should bear in mind that the terms of those

> explicit contracts can suffer interference from subsequent interpretation of events. Courts are *not* barred from concluding, and trading partners are *not* barred from arguing, that a *course of behavior* between electronic trading partners gives rise to an implicit legally enforceable agreement, or an implicit enforceable change to an explicit electronically-formed contract, even in the absence of further exchanges of legally binding messages.

The next section describes a pattern that may be used to explicitly exchange a series of one or more *transactions,* within a *collaboration,* to form a legally binding contract.

5.2 Simple contract formation pattern

Contracts MAY be formed by ebXML *collaborations* by the inclusion of *offers* and *acceptances* that conform to the Simple Contract Formation Pattern described here. This section describes a pattern that may be used to explicitly exchange a series of one or more *transactions,* within a *collaboration,* to form a legally binding contract. The Simple Contract Formation Pattern is constrained by rules that define a constrained subset of the alternative methods available for forming a contract under the ebBPSS schema. The pattern illustrates a subset of functionality that a particular domain or group of trading partners might elect.

5.2.1 Requirements for all business documents and document envelopes

To use this sample pattern, a business process must conform to the following rules, which are elective ("non-normative") to the ebBPSS standard, but required by this pattern:

1. Guard expressions in this pattern MUST refer only to one or more data fields that reside within the *Business Document* contained in the *Document Envelope* being evaluated. For example, this rules out a success or failure end state being generated by guard expressions that rely on the *Document Envelope* name, or the isPositiveResponse attribute of the *Document Envelope*.

2. *Business Documents* in this pattern MUST NOT set the IsLegallyBinding attribute to "No". This simplifies the evaluation that each business service interface must conduct of a document. Among other things, this rule also bars a number of approaches, such as the negotiating function demonstrated in the Simple Negotiation Pattern described in Section 6 of this document.

3. All *Business Transactions* and *Business Documents* in this pattern MUST conform to the one of the six "transaction patterns" defined in Chapter [9] of the UMM N90 metamodel. This is an example of re-use. The six recommended N90 patterns dictate or constrain the use of certain ebBPSS *business signal parameters* such as timeToPerform and timeToAcknowledgeReceipt. By re-using well-defined permutations of the *business signal parameter* values, the process designer and the process user can choose to rely on the UMM

N90 standard designers, who have in the UMM documentation described the logical relationship between the signals, and made suggestions about the suitability of particular permutations to particular business needs.

5.2.2 Requirements for all offers

Under this pattern:

1. A *document* constituting an *offer* MUST be the *Business Document* sent within the Requesting Business Activity.

2. Any *Business Document* constituting an *offer* MUST NOT contain any data that is evaluated by a guard expression but is not transmitted with the *Document Envelope* that contains that *Business Document*. Another way of putting this is that the offer document may not incorporate data by reference that would not be captured by an archive of the message in which the document is sent and received. (While it certainly may be possible for trading partners to work out an acceptably safe protocol for incorporation by linking reference, that function would make more complex the archiving of contract formation evidence. This simple pattern prohibits the linking so as to keep those archiving requirements very simple.

5.2.3 Requirements for all acceptances

Under this pattern:

1. Business processes MUST define one and only one responding *Business Document* that is evaluated by the processes' guard expressions as producing a "success" end state (and thus the end of that atomic *transaction*). That *document* constitutes the *acceptance, and* MUST be the *Business Document* sent within the *Responding Business Activity* of the same *Business Transaction* in which the *offer* was sent as the *Requesting Business Activity.*

2. Repeating the terms of an *offer*, in the *document* constituting an *acceptance* to that *offer*, is NOT RECOMMENDED. Repetition of terms previously transmitted creates ambiguity. If the terms sent "as accepted" are identical to those sent "as offered", a comparison by the offering party is redundant. The parties have already made provision for the desired level of message integrity and security by setting the business signal parameters. Therefore it is possible that the parties are already reflecting back acknowledgement messages. If the comparison reveals a difference, the comparing party is faced with ambiguity among the artifacts that might be its legally relevant evidence, and no clear rule for whether the document type or the document contents govern.

5.2.4 Requirements for all rejections and counteroffers

5.2.4.1 Handling of explicit substantive rejections

Under this pattern:

1. A *document* constituting a *rejection* MUST be the Business Document sent within the Responding Business Activity of the same Business Transaction in which the *offer* was sent as the Requesting Business Activity.

2. A *document* constituting a *rejection* terminates the *transaction* initiated by the *offer* being rejected, by transitioning to a "failure" end state.

5.2.4.2 Handling of counteroffers

The request-response paradigm of the BPSS (as well as the UMM N90 "transaction patterns" requires that all counteroffers be expressed in two documents or signals: (a) a *rejection*, to properly close the request-response pair initiated by the *offer*, and (b) a counteroffer, expressed as a new *offer* in which the rejecting party is the initiator of a new *transaction*.

Thus, under this pattern:

In order to propose new or modified terms, the rejecting party MUST send a new *offer* containing the proposed terms, thereby starting a new *transaction* response-request pair.

A *document* constituting a *rejection* MAY be bound to a signal indicating that a counteroffer is coming, which is called a "*counteroffer advice*" in this document.

A *counteroffer advice* MUST NOT be treated by itself as an *offer,* nor as a *binding document*.

A *counteroffer advice* MAY be communicated by a message *document* bound to the *rejection document* in a manner compliant with ebXML standards (such as in a common *Document Envelope*), or by a unique *rejection document* subtype used only to signify a *counteroffer advice* as well as a *rejection*. However, the method of indicating a *counteroffer advice* MUST be specified in the applicable CPA.

Receipt of a *counteroffer advice* MUST NOT toll or re-set a *transaction* time-out clock (such as timeToPerform) started by the rejected *offer*. The business service interface of an ebXML user MAY use the *counteroffer advice* for its own purposes.

It is RECOMMENDED that a *collaboration* handling system include a separate collaboration-oriented time-out clock, distinct from the ebBPSS timeToPerform rules applicable to an individual *transaction*. The rules for that clock may include an explicit manner for handling *counteroffer advice* messages. Under ebBPSS the time-out conclusions of that timer do not directly affect the timer objects in the schema's metamodel. However, it would likely inform the decisions of a business service interface decisions regarding, among other things, when to throw an explicit *rejection*, and when to rescind an *offer* (if the conditions of the *offer* permit it).

A separate *document* type for offers not capable of a counteroffer -- sometimes called "unalterable" offers -- is NOT RECOMMENDED. Under the ebBPSS schema, every *offer* must be simply *accepted* or *rejected* on a "take it or leave it" basis. Processing of counteroffers generally will be handled in a more robust and informative manner by the recipient's business

service interface interpreting the rejection, not by a preemptive failure caused by a *document* type.

A sidebar: The utility of patterns in handling business signal parameters

As standards that attempts to permit interoperability with a wide range of current practices, ebXML's schemas almost certainly provide more functionality than most users will initially employ. The BPSS schema specifies some mandatory signals and state handling functions, and many more optional ones. Some potential users may wish to permit or support only a select subset. Some user domains may wish to provide a simple upgrade path, by constraining their use of the BPSS schema parameters to a subset that maps easily to the cognate functions of their legacy system.

The Simple Contract Formation Pattern is an illustrative example of a set of rules that might be voluntarily adopted to present a simpler set of process design options. This is a hypothetical pattern, not an actual recommendation of suitability. It merely illustrates how a process designer might further constrain the possible uses of BPSS functionality to make it more "user-friendly" to a particular user base. As a result, a process designer could (1) offer to this use base only business processes that conform to the pattern, and (2) advise users to interrogate new business processes to see if they require functionalities that this pattern excludes.

5.3 Drop ship business process example

The following table illustrates the composition of a multiparty *collaboration* from multiple binary *collaborations* and *Business Transactions*, each composed of one or two *Business Documents*. This collaboration can be conducted under the Simple Contract Formation Pattern defined in the previous section. The UMM N90 transaction pattern applicable to each *transaction* is noted in brackets in the second column in the following table. The hypothetical *collaboration* is a superset of the same *Business Transactions* used as the illustrative values that populate the sample "Worksheets" in the ebXML Business Process Analysis Worksheet and Guidelines [bpWS].

DROP SHIP SCENARIO

SAMPLE USE OF BUSINESS PROCESS PATTERNS[1]

Version 1

10 May 2001

Jamie Clark, Bob Haugen, Nita Sharma, Dave Welsh, Brian Hayes

BUSINESS PROCESS	BINARY COLLABOR-ATION (protocol)	BUSINESS TRANSACTION (activity) [Pattern per N090]2	INITIATING / REQUESTING SIDE	REQUESTING DOCUMENT	RESPONDING SIDE	RESPONDING DOCUMENT
BPUC-5.7-Sales-Product-Notification Actors: Retailer, DSVendor	BC-6.9-Sales-Product-Offering	BT-8.9-Product-Offering *[Request / Confirm]*	PARTNER TYPE: DSVendor AUTH ROLE: Catalog Publishing	Product Catalog Offering (e.g. X12 832, ver 4010)	PARTNER TYPE: Retailer AUTH ROLE: Merchandising	Product Catalog Acceptance

[1] Notes on use of roles: Authorized Roles are assigned to each of the two roles in each Business Transaction. Each MUST be unique within a Business Process (or else you can't definitively point to them for process specification purposes). It is RECOMMENDED that Authorized Roles be named to facilitate resource discovery, by creating unique composite values from a controlled vocabulary. There is no normative rule for generating the names. In this table, we have used a *hypothetical* controlled vocabulary which includes *"Inventory Buyer, Catalog Publisher, Merchandising, Buying Customer, Customer Service, Accounts Receivable, , Shipper, , , Payer, Payee, , , Credit Authority Service, , "* , to promote resource discovery and re-use, and we have elected to use the Business Transaction names (and, where necessary, Collaboration names) to qualify and distinguish them.

[2] This column suggests use of one of the six demonstrative signal patterns offered in the UN/CEFACT TMWG N90 metamodel. Re-using these reduces our need to pay attention to the parameter values.

BUSINESS PROCESS	BINARY COLLABOR-ATION (protocol)	BUSINESS TRANSACTION (activity) [Pattern per N090]2	INITIATING / REQUESTING SIDE	REQUESTING DOCUMENT	RESPONDING SIDE	RESPONDING DOCUMENT
BPUC-5.6-Inventory-Management Actors: Retailer, DSVendor	BC-6.7-Vendor-Inventory-Reporting	BT-8-5-Vendor-Inventory-Report *[Notification]*	PARTNER TYPE: DSVendor AUTH ROLE: Inventory Buyer	Inventory Report	PARTNER TYPE: Retailer AUTH ROLE: Inventory Buyer	On Hand Product Availability
BPUC-5.1-Firm-Sales-Order Actors: Customer, Retailer	BC-6.1-Create-Customer-Order[3]	BT-8.1-Firm-Customer-Sales-Order *[Business Transaction]*	PARTNER TYPE: Customer AUTH ROLE: Buying Customer	Sales Order[4]	PARTNER TYPE: Retailer AUTH ROLE: Customer Service	Confirmation[5]
BPUC-5.2-Customer-Credit-Inquiry Actors: Retailer, Credit Authority	BC-6.2-Check-Customer-Credit[6]	BT-8.2-Check-Customer-Credit *[Request / Response]*	PARTNER TYPE: Retailer AUTH ROLE: Customer Service	Credit Check	PARTNER TYPE: Credit Authority AUTH ROLE: Credit Service	Credit Check Response

[3] In designing the business process, Retailer might choose to confirm the order only after successfully completing the *Product Fulfillment* collaboration. In that case *Order Fulfillment* would nest inside *Firm Order*.

[4] Provided via web browser.

[5] Provided via email

[6] The suggested pattern is "Request/Response", not "Commercial Transaction" in N90 usage, because information was transmitted on demand, but no economic commitment (credit allocation) was made.

BUSINESS PROCESS	BINARY COLLABOR-ATION (protocol)	BUSINESS TRANSACTION (activity) [Pattern per N090]2	INITIATING / REQUESTING SIDE	REQUESTING DOCUMENT	RESPONDING SIDE	RESPONDING DOCUMENT
BPUC-5.4-Purchase-Order-Management Actors: Retailer, Vendor	BC-6.4-Create-Vendor-Purchase-Order	BT-8.4-Create-Vendor-Purchase-Order *[Business Transaction]*	PARTNER TYPE: Retailer AUTH ROLE: Inventory Buyer	Purchase Order Request	PARTNER TYPE: DSVendor AUTH ROLE: Customer Service	Purchase Order Acknowledgment
BPUC-5.5-Ship-Goods Actors: DSVendor, Transport Carrier	BC-6.5-Shipment-Instruction	BT-8.7-Shipment-Notification *[Business Transaction]*	PARTNER TYPE: DSVendor AUTH ROLE: Shipper	Shipment Instruction	PARTNER TYPE: Transport Carrier AUTH ROLE: Customer Service	Bill of Lading
	BC-6.6-Confirm-Shipment	BT-8.8-Confirm-Shipment *[Notification]*	PARTNER TYPE: DSVendor AUTH ROLE: Shipper	Advance Ship Notice	PARTNER TYPE: Retailer AUTH ROLE: Customer Service	NONE
BPUC-5.3-Customer-Credit-Payment Actors: Retailer, Credit Authority	BC-6.3-Process-Credit-Payment	BT-8.3-Charge-Customer-Credit *[Business Transaction]*	PARTNER TYPE: Retailer AUTH ROLE: Accounts Receivable	Charge Credit	PARTNER TYPE: Credit Authority AUTH ROLE: Credit Authority Service	Confirm Credit
BPUC-5.8-Present-Invoice	BC-6.10-Invoice-Presentment	BT-8.11-Present-Invoice *[Notification]*	PARTNER TYPE: DSVendor AUTH ROLE: Payee	Invoice	PARTNER TYPE: Retailer AUTH ROLE: Payor	NONE

Table 6-1: Inventory of Key Objects for Drop Ship Hypothetical MultiParty Collaboration

6 Simple Automated Contract Negotiation in ebXML

6.1 ebBPSS contract negotiation functionality

In the prior section we examined contract formation by exchange of explicit, binding terms. At each step of the message exchange, the trading partners were making commitments that might (if properly met with a valid response) result in a "success" end state associated with an explicit contract formed by matching offer and acceptance.

Trading partners may also wish to exchange proposed terms, without making an assertion of intent to be legally bound. This is analogous to the paper contracting practice of exchanging unsigned drafts or term sheets.

Of course, trading parties may interrogate proposed business processes in a CPP or CPA independently, and then communicate in a human-readable fashion about the suitability and desirability of the specified process.

Under the ebBPSS, trading partners also have the opportunity to exchange Business Documents in a run-time fashion, with their isLegallyBinding parameter set to "No", and thereby test whether a particular sequence of exchanged BusinessDocuments results in a mutually satisfactory outcome.

Having done so, and concluded (independently) that the resulting collaboration is acceptable, the same partners are then in a position to efficiently duplicate the sequence by changing one parameter -- setting the isLegallyBinding parameter set to "Yes" throughout -- and thereby communicate the "dry run" contractual sequence as an enforceable transaction.

The generalized flow of events resulting from the foregoing approach is illustrated in the following activity diagram.

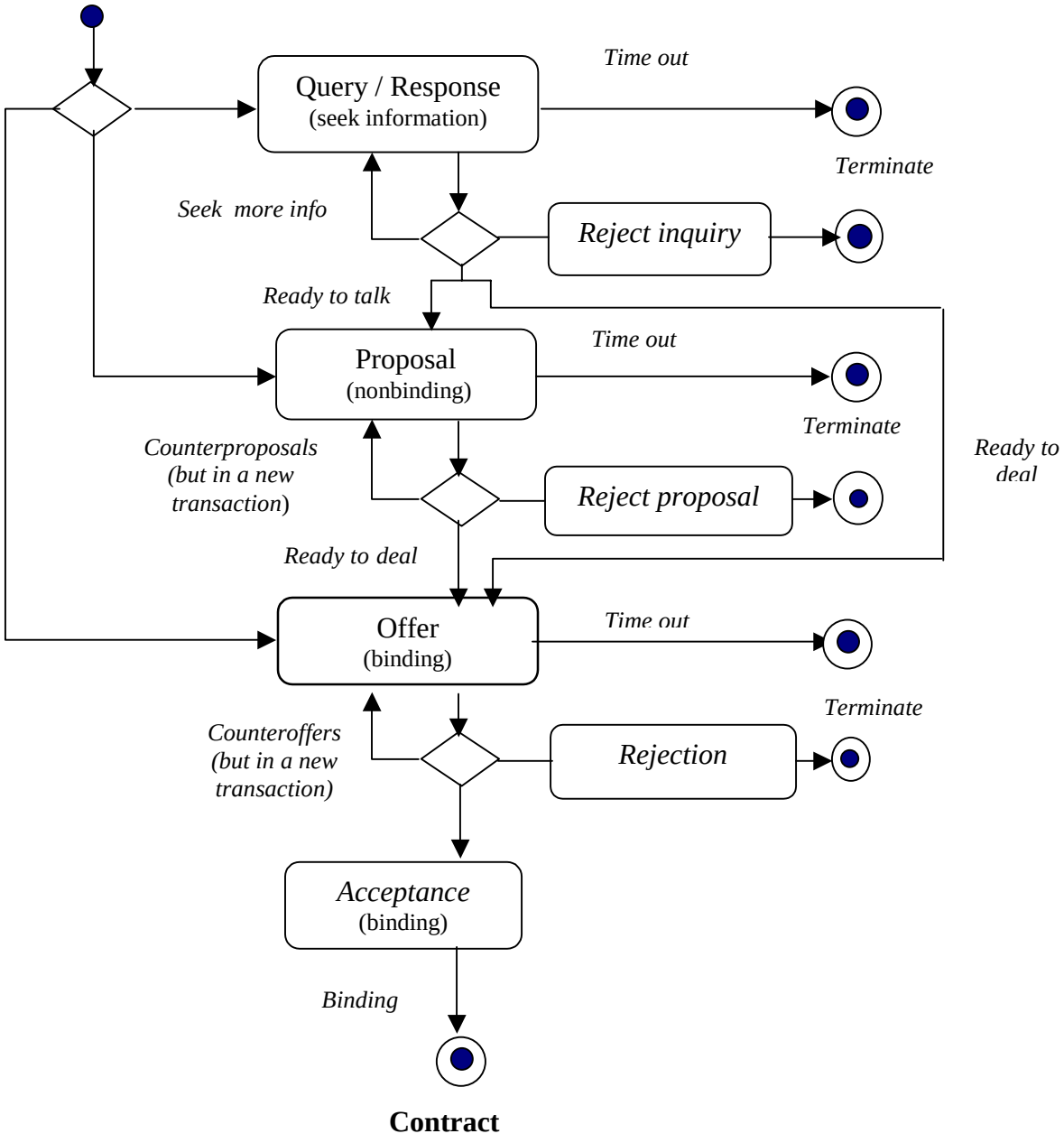

Figure 6-1: Hybrid Activity Diagram for Simple Negotiation Pattern

6.2 CPA negotiation as an instance

Some ebXML users may initiate communications by selecting from a sheaf of pre-set CPAs. Others may wish to negotiate a CPA dynamically by negotiating a choice from among a pre-set group of CPAs, or assembling a CPA from two CPPs. The Simple Negotiation Pattern may be used to perform such a negotiation, by sending a proposed CPA on a nonbinding basis (isLegallyBinding="No") as a BusinessDocument to a proposed trading partner, in a single BusinessTransaction which indicates that the sole guard expression condition for a "success" end state is return of the identical BusinessDocument, followed (consistent with the foregoing pattern) by either:

1. A nonbinding substantive acceptance, indicated by the return of the CPA, which can then be formally agreed by a second similar exchange with the isLegallyBinding parameter="Yes".

2. A rejection by explicit message, timeout or counteroffer advice, and in the latter case, a new exchange based on the CPA contained in the new offer heralded by the counteroffer advice.

The CPA Specification [ebCPP] requires signature of the CPA for substantive reasons. In order to satisfy that requirement, in the design of the foregoing process, the BusinessDocument containing the proposed CPA MUST bear a "isNonrepudiationOfReceiptRequired" parameter="Yes".

In order to initiate an ebXML compliant transaction, trading partners must refer to a CPA. If potential trading partners are attempting to negotiate a CPA in such a transaction, they MUST nevertheless agree to a common CPP under which the CPA negotiation occurs. It is RECOMMENDED that the prospective trading partner who initiates that preliminary negotiation do so by specifying agreement to a CPP already offered by the non-initiating party (e.g., held out in a registry as being available for that party).

Potential trading partners who wish to be assured that their negotiation over competing prospective CPAs will computationally resolve to a CPA, without human intervention, may choose to employ the suggested set of default business rules described in the "Conflict resolution of equally weighted options" section of the [Automatic CPA Negotiation] document. However, parties are free to accept or reject the adoption of those rules.[7]

[7]. Readers should note that the architects of the ebXML patterns *generally* seek to leave the selection of such matters up to the individual user. If I want to specify in a registry that I only transact in cuneiform on clay tablets, albeit wrapped in an ebXML data structure, the *standards* generally leave me free to do so. (As a practical matter, under the BPSS we would be looking at a "Business Document" constituting a conventional XML wrapper around a highly unconventional "Attachment". Also, to remain in compliance with the BPSS one would have to convert the cuneiform to transmittable form -- perhaps by shipping a JPEG file -- and setting the "spec" parameter of the "Attachment" object to a resolvable URI that allegedly informs a reader how to interpret the JPEG picture.) How the *market* may react to this is an entirely separate consideration.

7 Disclaimer

The views and specification expressed in this document are those of the authors and are not necessarily those of their employers. The authors and their employers specifically disclaim responsibility for any problems arising from correct or incorrect implementation or use of this design.

8 Contact Information

Team Leader (of the CC/BP Analysis group of the Joint Delivery Team):

Brian Hayes

Commerce One

4440 Rosewood Drive

Pleasanton, CA

USA

+1 (925) 788-6304

brian.hayes@commerceone.com

Editor

James Bryce Clark

McLure-Moynihan, Inc.

28720 Canwood Street Suite 208

Agoura Hills, CA 91301

USA

+1 (818) 706-3882

Jamie.clark@mmiec.com