



Creating A Single Global Electronic Market

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34

ebXML Business Process Specification Schema Version 1.0

Context/Metamodel Group
of the CC/BP Joint Delivery Team

27 April 2001

1 Status of this Document

This document is a final DRAFT for the *eBusiness* community. Distribution of this document is unlimited. This document will go through the formal *Quality Review* Process as defined by the *ebXML Requirements Document*. The formatting for this document is based on the Internet Society's Standard RFC format.

This version:

EbXML_BPschema_1.0

Latest version:

EbXML_BPschema_0.99

Previous version:

EbXML_BPschema_0.90

35 **2 ebXML BP/CoreComponents metamodel participants**

36 We would like to recognize the following for their significant participation to the development of
37 this document.

38

39 Team Lead:

40 Paul Levine, Telcordia

41

42 Editors:

43 Jim Clark, E2Open - previously Edifecs: (Transaction Semantics)

44 Cory Casanave, Data Access Technologies: (UML model)

45 Kurt Kanaskie, Lucent Technologies: (DTD and Examples)

46 Betty Harvey, Electronic Commerce Connection: (DTD documentation)

47 Jamie Clark, Spolin Silverman & Cohen LLP: (Legal aspects)

48 Neal Smith, Chevron: (Issues Lists, and W3C schema)

49 John Yunker, Edifecs: (Signal structures)

50 Karsten Riemer, Sun Microsystems: (Overall Document)

51

52 Participants:

53 Antoine Lonjon, Mega

54 J.J. Dubray, Excelon

55 Bob Haugen, Logistical Software

56 Bill McCarthy, Michigan State University

57 Paul Levine, Telcordia

58 Brian Hayes, CommerceOne

59 Nita Sharma, Netfish

60 David Welsh, Nordstrom

61 Christopher Ferris, Sun Microsystems

62 Antonio Carrasco, Data Access Technologies

63

64

65	3	Table of Contents	
66	1	Status of this Document	i
67	2	ebXML BP/CoreComponents metamodel participants	ii
68	3	Table of Contents	iii
69	4	Introduction	v
70	4.1	Summary of Contents of Document	1
71	4.2	Audience	1
72	4.3	Related Documents	1
73	4.4	Prerequisites	2
74	5	Design Objectives	2
75	5.1	Goals/Objectives/Requirements/Problem Description	2
76	5.2	Caveats and Assumptions	3
77	5.2.1	Relationship between <i>ebXML Business Process Specification Schema</i> and UMM	3
78	6	System Overview	5
79	6.1	Key Concepts of the ebXML Business Process Specification Schema	10
80	6.2	How to use the ebXML Business Process Specification Schema	14
81	6.3	How ebXML Business Process Specification Schema is used with other ebXML	
82		specifications	14
83	6.4	How to design collaborations and transactions, re-using at design time	16
84	6.4.1	Specify a Business Transaction and its Business Document Flow	16
85	6.4.2	Specify a Binary Collaboration	22
86	6.4.3	Specify a MultiParty Collaboration	25
87	6.4.4	Specify a Choreography	27
88	6.4.5	The whole model	31
89	6.5	Core Business Transaction Semantics	33
90	6.5.1	Interaction Predictability	33
91	6.5.2	Creating legally binding contracts	36
92	6.5.3	Non-Repudiation	37
93	6.5.4	Authorization security	38
94	6.5.5	Document security	39
95	6.5.6	Reliability	40
96	6.5.7	Parameters required for CPP/CPA	40
97	6.6	Run time Business Transaction semantics	40
98	6.6.1	Timeouts	41
99	6.6.2	Exceptions	43
100	6.7	Runtime Collaboration Semantics	46
101	6.8	Where the ebXML Business Process Specification Schema May Be Implemented	46
102	7	UML Element Specification	46
103	7.1	Business Collaborations	46
104	7.1.1	MultiPartyCollaboration	46
105	7.1.2	BusinessPartnerRole	47
106	7.1.3	Performs	47
107	7.1.4	AuthorizedRole	48
108	7.1.5	BinaryCollaboration	48

109	7.1.6 BusinessActivity	50
110	7.1.7 BusinessTransactionActivity	50
111	7.1.8 CollaborationActivity	51
112	7.2 Business Transactions	51
113	7.2.1 BusinessTransaction	51
114	7.2.2 Business Action	52
115	7.2.3 RequestingBusinessActivity	53
116	7.2.4 RespondingBusinessActivity	54
117	7.3 Document flow	55
118	7.3.1 Document Security	55
119	7.3.2 Document Envelope	55
120	7.3.3 BusinessDocument	56
121	7.3.4 DocumentSpecification	57
122	7.3.5 Attachment	57
123	7.4 Choreography within Collaborations	58
124	7.4.1 BusinessState	58
125	7.4.2 Transition	59
126	7.4.3 Start	59
127	7.4.4 CompletionState	60
128	7.4.5 Success	60
129	7.4.6 Failure	61
130	7.4.7 Fork	61
131	7.4.8 Join	61
132	7.5 Definition and Scope	62
133	7.6 Collaboration and transaction well-formedness rules	62
134	8 ebXML Business Process Specification Schema – (DTD)	65
135	8.1 Documentation for the DTD	65
136	8.2 XML to UML cross-reference	92
137	8.3 Scoped Name Reference	93
138	8.4 Sample XML document against above DTD	95
139	9 Business signal structures	95
140	9.1.1 ReceiptAcknowledgment DTD	95
141	9.1.2 AcceptanceAcknowledgement DTD	97
142	9.1.3 Exception Signal DTD	99
143	10 Production Rules	101
144	Appendix A: Sample XML Business Process Specification	103
145	Appendix B: Business Process Specification Schema DTD	108
146	Appendix C: Business Process Specification Schema XML Schema	115
147	11 References	126
148	12 Disclaimer	126
149	13 Contact Information	127
150		

151 **4 Introduction**

152 **Executive Summary**

153

154 The ebXML Specification Schema provides a standard framework by which business
155 systems may be configured to support execution of business collaborations consisting of
156 business transactions. It is based upon prior UN/CEFACT work, specifically the
157 metamodel behind the UN/CEFACT Modeling Methodology (UMM) defined in the
158 N090R9.1 specification.

159 The Specification Schema supports the specification of Business Transactions and the
160 choreography of Business Transactions into Business Collaborations. Each Business
161 Transaction can be implemented using one of many available standard patterns. These
162 patterns determine the actual exchange of Business Documents and business signals
163 between the partners to achieve the required electronic commerce transaction.

164 The current version of the specification schema addresses collaborations between two
165 parties (Binary Collaborations).

166 It is anticipated that a subsequent version will address additional features such as the
167 semantics of economic exchanges and contracts, more complex multi-party
168 choreography, and context based content.

169 **4.1 Summary of Contents of Document**

170 This document describes the ebXML Specification Schema

171 This document describes the Specification Schema, both in its UML form and in
172 its DTD form.

173 The document first introduces general concepts and semantics, then applies
174 these semantics in a detail discussion of each part of the model. The document
175 then specifies all elements in the UML form, and then in the XML form.

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

179

180 **4.2 Audience**

181 The primary audience is business process analysts. We define a business
182 process analyst as someone who interviews business people and as a result
183 documents business processes in unambiguous syntax.

184 An additional audience is designers of business process definition tools who
185 need to specify the conversion of user input in the tool into the XML
186 representation of the Specification Schema.

187 The audience is not business application developers.

188 **4.3 Related Documents**

189 As mentioned above, other documents provide detailed definitions of some of the
190 components of the ebXML Specification Schema and of their inter-relationship.
191 They include ebXML Specifications on the following topics:

192

193 • ebXML Technical Architecture, version 1.0

194 • ebXML Core Components, version 1.0

195 • ebXML Naming Convention for Core Components, version 1.0

196 • ebXML Core Component and Business Process Document Overview,
197 version 1.0

198 • ebXML Collaboration-Protocol Profile and Agreement Specification V0.95

199 • ebXML Business Process and Business Information Analysis Overview,
200 version 0.7

201 • ebXML Business Process Analysis Worksheets & Guidelines, version
202 0.99

203 • ebXML E-Commerce Patterns, version 0.99

204 • ebXML Catalog of Common Business Processes, version 0.99

- 205 • ebXML Message Service Specification V0.99
- 206 • UN/CEFACT Modeling Methodology (UMM) as defined in the N090R9.1
- 207 specification

208 **4.4 Prerequisites**

209 It is assumed that the audience will be familiar with or have knowledge of the
210 following technologies and techniques:

- 211 • Business process modeling techniques and principles
- 212 • The UML syntax and semantics
- 213 • The Extensible Markup Language (XML)

214 **5 Design Objectives**

215 **5.1 Goals/Objectives/Requirements/Problem Description**

216 Business process models describe interoperable business processes that allow
217 business partners to collaborate. Business process models for e-business must
218 be turned into software components that collaborate on behalf of the business
219 partners.

220 The goal of the ebXML Specification Schema is to provide the bridge between e-
221 business process modeling and specification of e-business software
222 components.

223 The ebXML Specification Schema provides for the nominal set of specification
224 elements necessary to specify a collaboration between business partners, and to
225 provide configuration parameters for the partners' runtime systems in order to
226 execute that collaboration between a set of e-business software components.

227 A specification created against the ebXML Business Process Specification
228 Schema is referred to as an ebXML Business Process Specification.

229 The *ebXML Business Process Specification Schema* is available in two stand-
230 alone representations, a UML version, and an XML version.

231 The UML version of the *ebXML Business Process Specification Schema* is
232 merely a UML Class Diagram. It is not intended for the direct creation of ebXML
233 Business Process Specifications. Rather, it is a self-contained statement of all
234 the specification elements and relationships required to be able to create an
235 ebXML compliant Business Process Specification. Any methodologies and/or
236 metamodels used for the creation of ebXML compliant Business Process
237 Specifications must at minimum support these elements and relationships.

238 The XML version of the *ebXML Business Process Specification Schema* provides
239 the specification for XML based instances of ebXML Business Process
240 Specifications, and as a target for production rules from other representations.
241 Both a DTD and a W3C Schema is provided.

242 The UML and XML based versions of the *ebXML Business Process*
243 *Specification Schema* are unambiguously mapped to each other.

244 **5.2 Caveats and Assumptions**

245 This specification is designed to specify the run time aspects of a business
246 collaboration.

247 It is not intended to incorporate a methodology, and does not directly prescribe
248 the use of a methodology. However, if a methodology is to be used, it is
249 recommended that it be UN/CEFACT Modeling Methodology (UMM).

250 The *ebXML Business Process Specification Schema* does not by itself define
251 Business Documents Structures. It is intended to work in conjunction with already
252 existing Business Document definitions, and/or the document metamodel defined
253 by the ebXML Core Components specifications.

254 **5.2.1 Relationship between *ebXML Business Process Specification*** 255 ***Schema* and UMM**

256
257 The UN/CEFACT Modeling Methodology (UMM) is a methodology for business
258 process and information modeling.

259 This section describes the relationship between UMM and the *ebXML Business*
260 *Process Specification Schema*.

261 The UMM Meta Model is a description of business semantics that allows Trading
262 Partners to capture the details for a specific business scenario (a Business
263 Process) using a consistent modeling methodology. A Business Process
264 describes in detail how Trading Partners take on shared roles, relationships and
265 responsibilities to facilitate interaction with other Trading Partners. The
266 interaction between roles takes place as a choreographed set of Business
267 Transactions. Each Business Transaction is expressed as an exchange of
268 electronic Business Documents. The sequence of the exchange is determined
269 by the Business Process, and by messaging and security considerations.
270 Business Documents are composed from re-useable Business Information
271 Objects. At a lower level, Business Processes can be composed of re-useable
272 Common Business Processes, and Business Information Objects can be
273 composed of re-useable Core Components. Common Business Processes and
274 Business Information Objects reside in a UMM Business Library.

275 The UMM Meta Model supports a set of Business Process viewpoints that
276 provide a set of semantics (vocabulary) for each viewpoint and forms the basis of
277 specification of the semantics and artifacts that are required to facilitate business
278 process and information integration and interoperability. Using the UMM
279 methodology and the UMM metamodel, the user may thus create a complete
280 Business Process and Information Model. This model contains more information
281 than what is required for configuring ebXML compliant software. Also the model
282 is syntax independent and not directly interpretable by ebXML compliant
283 software.

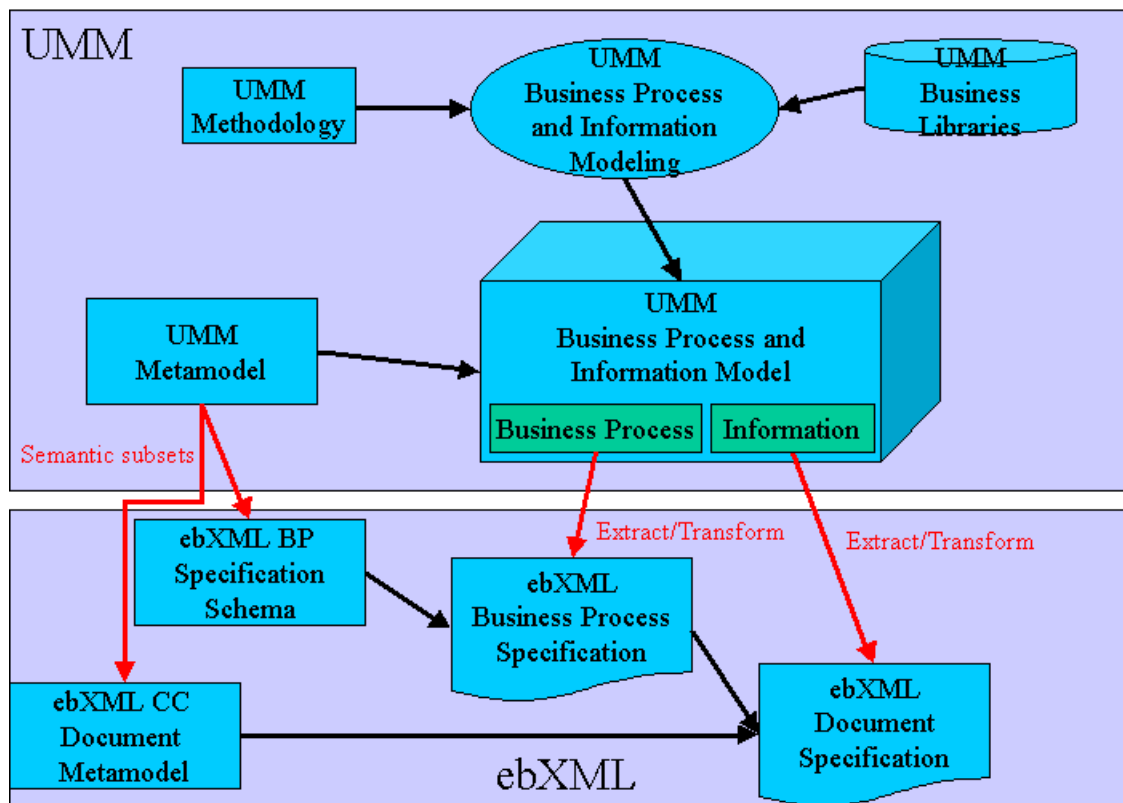
284 The *ebXML Business Process Specification Schema* provides an additional view
 285 of the UMM metamodel. This subset is provided to support the direct
 286 specification of the nominal set of elements necessary to configure a runtime
 287 system in order to execute a set of ebXML business transactions. By drawing
 288 out modeling elements from several of the other views, the *ebXML Business
 289 Process Specification Schema* forms a semantic subset of the UMM Meta Model.
 290 Using the *ebXML Business Process Specification Schema* the user may thus
 291 create a Business Process Specification that contains only the information
 292 required to configure ebXML compliant software.

293 The *ebXML Business Process Specification Schema* is available in two stand-
 294 alone representations, a UML version, and an XML version. The XML version is
 295 intended to be interpretable by ebXML compliant software.

296 The relationship between the UMM Meta Model and the *ebXML Business
 297 Process Specification Schema* is shown in Figure 1.

298 **Figure 1. UMM Metamodel and *ebXML Business Process Specification Schema***

299



300

301 Using the UMM methodology, and drawing on content from the UMM Business
 302 Library a user may create complete Business Process and Information Model
 303 conforming to the UMM metamodel.

304 Since the ebXML *Business Process Specification Schema* is a semantic subset
305 of the UMM metamodel, the user may then in an automated fashion extract from
306 the Business Process and Information Model the required set of elements and
307 relationships, and transform them into an ebXML Business Process Specification
308 conforming to the *ebXML Business Process Specification Schema*.

309 Likewise, since the ebXML CC document metamodel is aligned with the UMM
310 Metamodel, the user may then in an automated fashion extract from the
311 Business Process and Information Model the required set of elements and
312 relationships, and transform them into an ebXML document model conforming to
313 ebXML Core Component specifications.

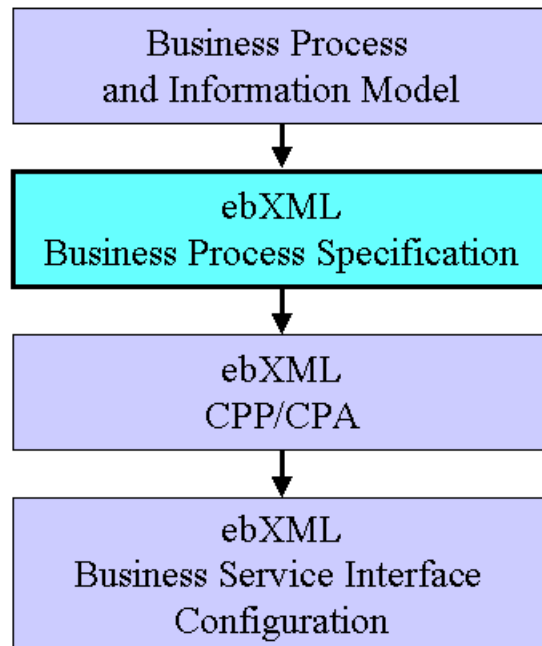
314 The UMM methodology is not part of the formal set of ebXML specifications.

315 Likewise, the UMM metamodel in its entirety is not part of the formal set of
316 ebXML specifications. Only the semantic subset represented by the *ebXML*
317 *Business Process Specification Schema* and CC are part of the formal set of
318 ebXML specifications.

319 The remainder of this document focuses on the *ebXML Business Process*
320 *Specification Schema* and Business Process Specifications created against it. It
321 is understood that proper Business Process and Information Modeling may have
322 taken place prior to beginning the activity of creating a Business Process
323 Specification.

324 **6 System Overview**

325 The ebXML *Business Process Specification Schema* provides a standard
326 framework for business process specification. As such, it works with the ebXML
327 Collaboration Protocol Profile (CPP) and Collaboration Protocol Agreement
328 (CPA) specifications to bridge the gap between Business Process Modeling and
329 the configuration of ebXML compliant e-commerce software, e.g. an ebXML
330 Business Service Interface, as depicted in Figure 2.



331

332 Figure 2: Business Process Specification and Business Service Interface Configuration

333 Using Business Process Modeling, a user may create a complete Business
334 Process and Information Model.

335 Based on this Business Process and Information Model and using the ebXML
336 *Business Process Specification Schema* the user will then extract and format the
337 nominal set of elements necessary to configure an ebXML runtime system in
338 order to execute a set of ebXML business transactions. The result is an ebXML
339 *Business Process Specification*.

340 Alternatively the ebXML *Business Process Specification* may be created directly,
341 without prior explicit business process modeling.

342 An ebXML *Business Process Specification* contains the specification of Business
343 Transactions and the choreography of Business Transactions into Business
344 Collaborations.

345 This ebXML *Business Process Specification* is then the input to the formation of
346 ebXML trading partner Collaboration Protocol Profiles and Collaboration Protocol
347 Agreements.

348 These ebXML trading partner Collaboration Protocol Profiles and Collaboration
349 Protocol Agreements in turn serve as configuration files for ebXML Business
350 Service Interface software.

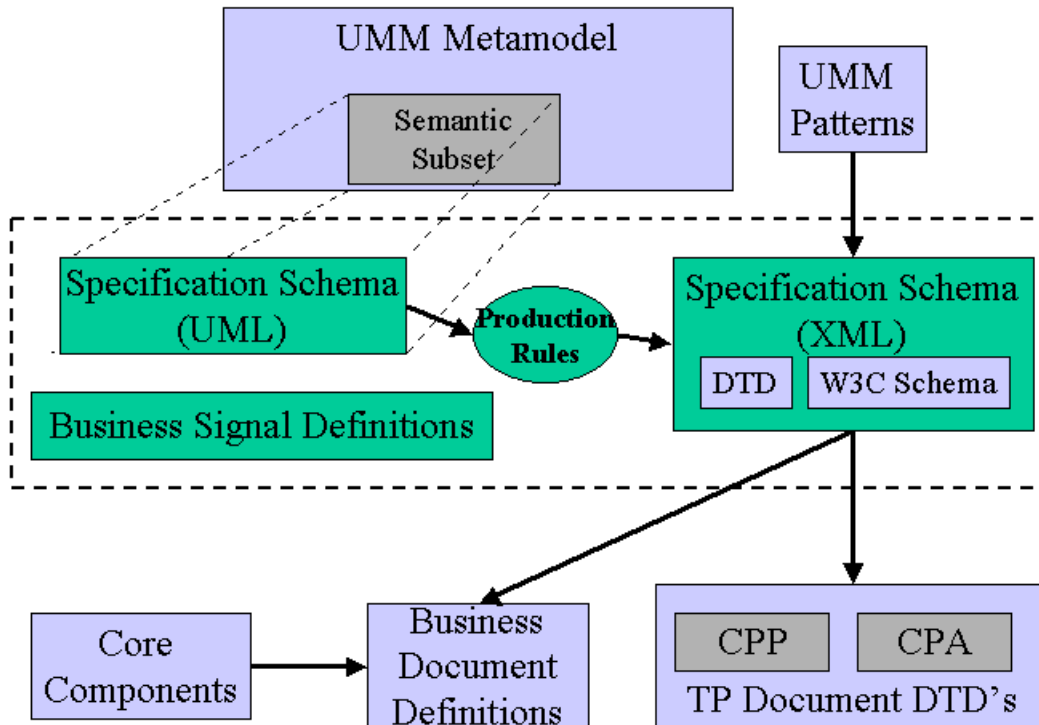
351 The architecture of the ebXML *Business Process Specification Schema* consists of
 352 the following functional components:

- 353 • UML version of the *Business Process Specification Schema*
- 354 • XML version of the *Business Process Specification Schema*
- 355 • Production Rules defining the mapping from the UML version of the
 356 *Business Process Specification Schema* to the XML version
- 357 • Business Signal Definitions

358

359 Together these components allow you to fully specify all the run time aspects of a
 360 business process model.

361 These components are shown (inside the dotted box) in Figure 3 below.
 362



363

364 **Figure 3: Relationship of ebXML Business Process Specification Schema to UMM, CPP/CPA**
 365 **and Core Components**

366

367 The following provides a description of each of the components in the ebXML
 368 *Business Process Specification Schema* and their relationship to UMM, and
 369 ebXML CC and CPP/CPA:

370 **UML version of Business Process Specification Schema**

371 The UML version of the *ebXML Business Process Specification Schema* is a
372 semantic subset of the metamodel behind UMM as specified in UN/CEFACT
373 TMWG's N090R9.1
374 N090R9.1 is as of this writing not yet approved by UN/CEFACT. It is the intent to
375 keep the *ebXML Business Process Specification Schema* and the UN/CEFACT
376 TMWG's N090 semantically aligned.

377 The UML version of the *ebXML Business Process Specification Schema* is
378 merely a UML Class Diagram. It is not intended for the direct creation of ebXML
379 Business Process Specifications. Rather, it is a self-contained statement of all
380 the specification elements and relationships required to be able to create an
381 ebXML compliant Business Process Specification.

382 **XML version of Business Process Specification Schema**

383 The XML version of the *ebXML Business Process Specification Schema* provides
384 the specification for XML based instances of *ebXML Business Process*
385 *Specifications*, and as a target for production rules from other representations.
386 Thus, a user may either create a *Business Process Specification* directly as an
387 XML document, or may chose to use some other means of specification first and
388 then apply production rules to arrive at the XML document version.

389 Any methodologies and/or metamodels used for the creation of ebXML compliant
390 Business Process Specifications must at minimum support the production of the
391 elements and relationships contained in the XML version of the *ebXML Business*
392 *Process Specification Schema*.

393 Both a DTD and a W3C Schema is provided. Each is an isomorphic definition of
394 the UML version of the *ebXML Business Process Specification Schema*.

395 **UMM Business Process Interaction Patterns**

396 ebXML Business Service Interfaces are configured to execute the business
397 processes specified in a *Business Process Specification*. They do so by
398 exchanging ebXML messages and business signals.

399 Each Business Transaction can be implemented using one of many available
400 standard patterns. These patterns determine the actual exchange of messages
401 and business signals between the partners to achieve the required electronic
402 commerce transaction.

403 The Business Transaction Interaction Patterns set forth in Chapter 8 of the UMM
404 N090R9.1 document illustrate recommended permutations of message
405 sequences as determined by the type of business transaction defined and the
406 timing policies specified in the transactions.

407 While the UMM patterns themselves are not part of the ebXML specifications, all
408 the security and timing parameters required to express the pattern properties are
409 provided as attributes of elements in the *ebXML Business Process Specification*
410 *Schema*.

411 ***Business Signal Definitions***

412 Business signals are application level documents that 'signal' the current state of
413 the business transaction. These business signals have specific business purpose
414 and are separate from lower protocol and transport signals.

415 However, the structures of ebXML business signals are 'universal' and do not
416 vary from transaction to transaction. Thus, they can be defined once and for all
417 as part of the ebXML *Business Process Specification Schema* itself.

418 The Business Process Specification Schema provides both the choreography of
419 business signals, and the structure definition of the business payload of a
420 business signal. The ebXML Message Service Specification signal structures
421 provide business service state alignment infrastructure, including unique
422 message identifiers and digests used to meet the basic process alignment
423 requirements. The business signal payload structures provided herein are
424 optional and normative and are intended to provide business and legal semantic
425 to the business signals

426 A DTD is provided for each of the possible business signals.

427 ***Production Rules***

428 A set of production rules are provided, defining the mapping from the UML
429 version of the ebXML *Business Process Specification Schema* to the XML
430 version.

431 The primary purpose for these production rules is to govern the one-time
432 generation of the DTD version of the ebXML *Business Process Specification*
433 *Schema* from the UML Class Diagram version of the ebXML *Business Process*
434 *Specification Schema*.

435 The Class Diagram version of *Business Process Specification Schema* is not
436 intended for the direct creation of ebXML Business Process Specifications.
437 However, if a *Business Process Specification* was in fact (programmatically)
438 created as an instance of this class diagram, the production rules would also
439 apply for its conversion into a DTD conformant XML document.

440 Separately, it is expected that a set of production rules will be constructed for the
441 production of an XML version of an ebXML *Business Process Specification* from
442 a set of UML diagrams constructed through the use of UMM.

443 An instance of the UML Class Diagram version of the ebXML *Business Process*
444 *Specification Schema* will through the application of its production rules produce
445 an XML Specification Document that is analytically, semantically and functionally
446 equivalent to one arrived at by modeling the same subset through the use of
447 UMM and its associated production rules.

448 ***Relationship to CPP/CPA***

449 A *Business Process Specification* is in essence the machine interpretable run
450 time business process specification needed for an ebXML Business Service
451 Interface. The *Business Process Specification* is therefore incorporated with or
452 referenced by ebXML trading partner Collaboration Protocol Profiles (CPP) and

453 Collaboration Protocol Agreements (CPA). Each CPP declares its support for
454 one or more Roles within the *Business Process Specification* . Within these CPP
455 profiles and CPA agreements are then added further technical parameters
456 resulting in a full specification of the run-time software at each trading partner.

457 ***Relationship to CC***

458 The *Business Process Specification Schema* does not by itself support the
459 definition of Business Documents. Rather, a *Business Process Specification*
460 merely points to the definition of Business Documents. Such definitions may
461 either be XML based, or – as attachments – may be any other structure, or
462 completely unstructured. XML based Business Document Specifications may be
463 based on the ebXML Core Components specifications.

464 ***Relationship to ebXML Message Service Specification***

465 The Business Process Specification Schema will provide choreography of
466 business messages and signals. The ebXML Message Service Specification
467 provides the infrastructure for message / signal identification, typing, and
468 integrity; as well as placing any one message in sequence with respect to other
469 messages in the choreography.

470

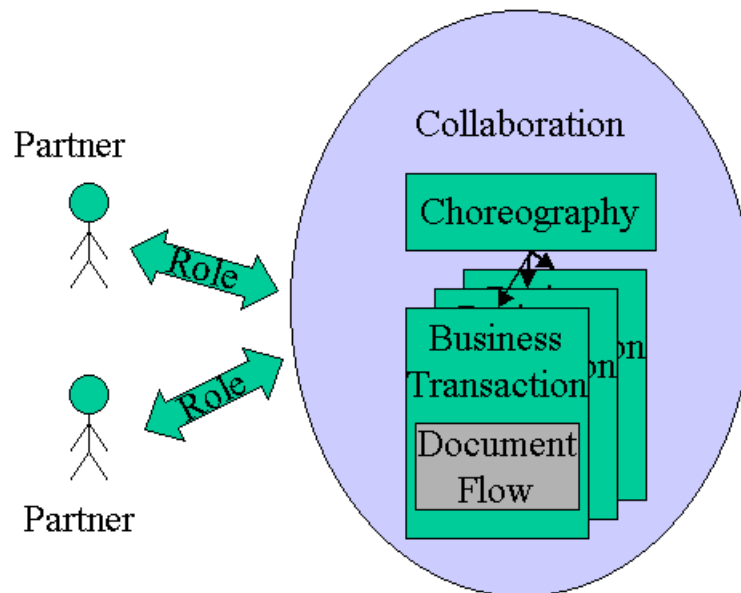
471 **6.1 Key Concepts of the ebXML Business Process Specification** 472 ***Schema***

473

474 The ebXML *Business Process Specification Schema* provides the semantics,
475 elements, and properties necessary to define business collaborations.

476 A business collaboration consists of a set of roles collaborating through a set of
477 choreographed transactions by exchanging business documents.

478 These basic semantics of a business collaboration are shown in Figure 4.



479

480

Figure 4. Basic Semantics of a business collaboration

481

482 Two or more business partners participate in the business collaboration through
 483 roles. The roles interact with each other through Business Transactions. The
 484 business transactions are sequenced relative to each other in a Choreography.
 485 Each Business Transaction consists of one or two predefined Business
 486 document flows. A Business Transaction may be additionally supported by one
 487 or more Business Signals.

488

The following section describes the concepts of a Business Collaboration, a
 489 Business Transaction, a Business document flow, and a Choreography

490

1. Business Collaborations

491

A business collaboration is a set of Business Transactions between
 492 business partners. Each partner plays one or more roles in the
 493 collaboration.

494

The ebXML *Business Process Specification Schema* supports two levels
 495 of business collaborations, Binary Collaborations and Multiparty
 496 Collaborations.

497

Binary Collaborations are between two roles only.

498 Multiparty Collaborations are among more than two roles, but such
499 Multiparty Collaborations are always synthesized from two or more Binary
500 Collaborations. For instance if Roles A, B, and C collaborate and all
501 parties interact with each other, there will be a separate Binary
502 Collaboration between A and B, one between B and C, and one between
503 A and C. The Multiparty Collaboration will be the synthesis of these three
504 Binary Collaborations.

505 Binary Collaborations are expressed as a set of Business Activities
506 between the two roles. Each Business Activity reflects a state in the
507 collaboration. The Business Activity can be a Business Transaction
508 Activity, i.e. the activity of conducting a single Business Transaction, or a
509 Collaboration Activity, i.e. the activity of conducting another Binary
510 Collaboration. An example of the former is the activity of placing a
511 purchase order. An example of the latter is the activity of negotiating a
512 contract. In either case the activities can be choreographed relative to
513 other activities as per below.

514 The ability of a Binary Collaboration to have activities that in effect are
515 executing other Binary Collaborations, is the key to recursive
516 compositions of Binary Collaboration, and to the re-use of Binary
517 Collaborations.

518 In essence each Binary Collaboration is a re-useable protocol between
519 two roles.

520 2. Business Transactions

521 A Business Transaction is the atomic unit of work in a trading
522 arrangement between two business partners. A Business Transaction is
523 conducted between two parties playing opposite roles in the transaction.
524 The roles are always a requesting role and a responding role.

525 Like a Binary Collaboration, a Business Transaction is a re-useable
526 protocol between two roles. The way it is re-used is by referencing it from
527 a Binary Collaboration through the use of a Business Transaction Activity
528 as per above. In a Business Transaction Activity the roles of the Binary
529 Collaboration are assigned to the execution of the Business Transaction.

530 Unlike a Binary Collaboration, however, the Business Transaction is
531 atomic, it cannot be decomposed into lower level Business Transactions.

532 A Business Transaction is a very specialized and very constrained
533 protocol, in order to achieve very precise and enforceable transaction
534 semantics. These semantics are expected to be enforced by the software
535 managing the transaction, i.e. an ebXML Business Service Interface
536 (BSI).

537 A Business Transaction will always either succeed or fail. If it succeeds it
538 may be designated as legally binding between the two partners", or
539 otherwise govern their collaborative activity. If it fails it is null and void,
540 and each partner must relinquish any mutual claim established by the
541 transaction. This can be thought of as 'rolling back' the Business
542 Transaction upon failure.

- 543 3. Business Document flows
- 544 A business transaction is realized as Business Document flows between
545 the requesting and responding roles. There is always a requesting
546 Business Document, and optionally a responding Business Document,
547 depending on the desired transaction semantics, e.g. one-way notification
548 vs. two-way conversation.
- 549 Actual document definition is achieved using the ebXML core component
550 specifications, or by some methodology external to ebXML but resulting in
551 a DTD or Schema that an ebXML *Business Process Specification* can
552 point to.
- 553 4. Choreography
- 554 The Business Transaction Choreography describes the ordering and
555 transitions between business transactions or sub collaborations within a
556 binary collaboration. In a UML tool this can be done using a UML activity
557 diagram. The choreography is described in the ebXML *Business Process*
558 *Specification Schema* using activity diagram concepts such as start state,
559 completion state, activities, synchronizations, transitions between
560 activities, and guards on the transitions.
- 561 5. Patterns
- 562 The ebXML *Business Process Specification Schema* provides a set of
563 unambiguous semantics within which to specify transactions and
564 collaborations. Within these semantics the user community has flexibility
565 to specify an infinite number of specific transactions and collaborations.
566 The use of predefined patterns combines this flexibility with a consistency
567 that facilitates faster design, faster implementation, and enables generic
568 processing.
- 569 A set of predefined transaction interaction patterns, defining common
570 combinations of transaction interaction parameter settings can be found
571 in UMM.
- 572 While the UMM transaction interaction patterns themselves are not part of
573 the ebXML specifications, all the security and timing parameters required
574 to express the pattern properties are provided as attributes of elements in
575 the *Business Process Specification Schema*.
- 576 It is also anticipated that patterns for collaboration choreographies will
577 emerge. An example of such a pattern is in the ebXML E-Commerce and
578 Simple Negotiation Patterns.
- 579 Re-use, recursion, and patterns are among the key concepts of the ebXML
580 *Business Process Specification Schema*. The following section will illustrate
581 these key concepts.

582 **6.2 How to use the ebXML Business Process Specification** 583 **Schema**

584 The ebXML *Business Process Specification Schema* should be used wherever
585 ebXML compliant software is being specified to execute Business
586 Collaborations. The generic term for such software is a Business Service
587 Interface (BSI).

588 The ebXML *Business Process Specification Schema* is used to specify the
589 business process related configuration parameters for configuring a BSI to
590 execute these collaborations.

591 This section discusses

- 592 • How the ebXML *Business Process Specification Schema* fits in with other
593 ebXML specifications.
- 594 • How to use the ebXML *Business Process Specification Schema* at design
595 time, either for specifying brand new collaborations and transactions, or
596 for re-using existing ones.
- 597 • How to specify core transaction semantics and parameters needed for a
598 Collaboration-Protocol Profile and Agreement (CPP/CPA).
- 599 • Run-time transaction and collaboration semantics that the ebXML
600 *Business Process Specification Schema* specifies and the Business
601 Service Interface (BSI) is expected to manage.

602 **6.3 How ebXML Business Process Specification Schema is** 603 **used with other ebXML specifications**

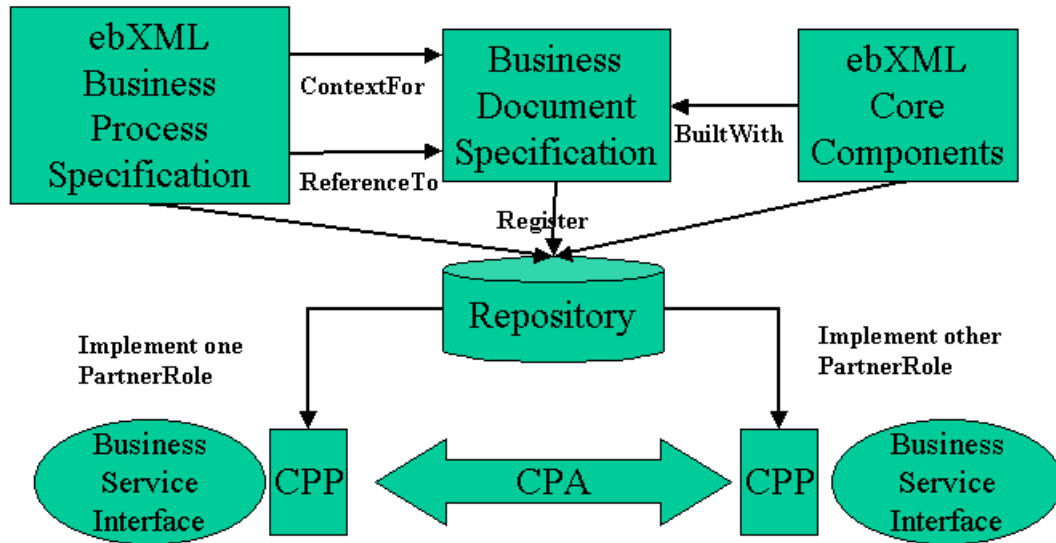
604 The ebXML *Business Process Specification Schema* provides the semantics,
605 elements, and properties necessary to define Business Collaborations.
606

607 A collaboration consists of a set of roles collaborating through a set of
608 choreographed transactions by exchanging Business Documents.

609 As shown in Figure 5, Business Documents are defined at the intersection
610 between the Business Process Specification and the ebXML Core Component
611 specifications. A Business Process Specification will reference, but not define, a
612 set of required Business Documents. At ebXML Business Documents are either
613 defined by some external document specification, or assembled directly or
614 indirectly from lower level information structures called core components. The
615 assembly is based on a set of contexts, many of which are provided by the
616 business processes, i.e. collaborations that use the documents in their document
617 flows.

618 The combination of the business process specification and the document
619 specification become the basis against which partners can make agreements on
620 conducting electronic business with each other.

621



622

623

Figure 5: ebXML Business Process Specification Schema and other ebXML Specifications

624

625

626

627

The user will extract and transform the necessary information from an existing Business Process and Information Model. Associated production rules could aid in creating an XML version of a *Business Process Specification*.

628

629

630

Alternatively a user would use an XML based tool to produce the XML version directly. Production rules could then aid in converting into XML, so that it could be loaded into a UML tool, if required.

631

632

633

634

In either case, the XML version of the *Business Process Specification* gets stored in the ebXML repository and registered in the ebXML registry for future retrieval. The *Business Process Specification* would be registered using classifiers derived during its design.

635

636

637

638

639

When implementers want to establish trading partner Collaboration Protocol Profile and Agreement the *Business Process Specification* XML document, or the relevant parts of it, are simply imbedded in or referenced by the CPP and CPA XML documents. ebXML CPP and CPA documents can only reference ebXML *Business Process Specifications* and only XML versions thereof.

640

641

Guided by the CPP and CPA specifications the resulting XML document then becomes the configuration file for one or more Business Service Interfaces (BSI),

642 i.e. the software that will actually manage either partner's participation in the
643 collaboration.

644 **6.4 How to design collaborations and transactions, re-using at** 645 **design time**

646

647 This section describes the ebXML *Business Process Specification Schema*
648 modeling relationships by building a complete Multiparty Collaboration from the
649 bottom up, as follows:

- 650 1. Specify a Business Transaction
- 651 2. Specify the Business Document flow for a Business Transaction
- 652 3. Specify a Binary Collaboration re-using the Business Transaction
- 653 4. Specify a Choreography for the Binary Collaboration
- 654 5. Specify a higher level Binary Collaboration re-using the lower level Binary
655 Collaboration
- 656 6. Specify a Multiparty Collaboration re-using Binary Collaborations

657 Although this section, for purposes of introduction, discusses the specification of
658 a collaboration from the bottom up, the ebXML *Business Process Specification*
659 *Schema* very much is intended for specifying collaborations from the top down,
660 re-using existing lower level content as much as possible.

661 The constructs listed above support the specification of fairly complex multi party
662 collaborations. However, an ebXML compliant Business Process Specificaton
663 may be as simple as a single Binary Collaboration referencing a single Business
664 Transaction. This involves only numbers 1 through 3 above. In other words,
665 Higher-level Binary Collaborations, Multi-party Collaborations and choreography
666 expressions are not required ebXML Business Process Specification compliance.

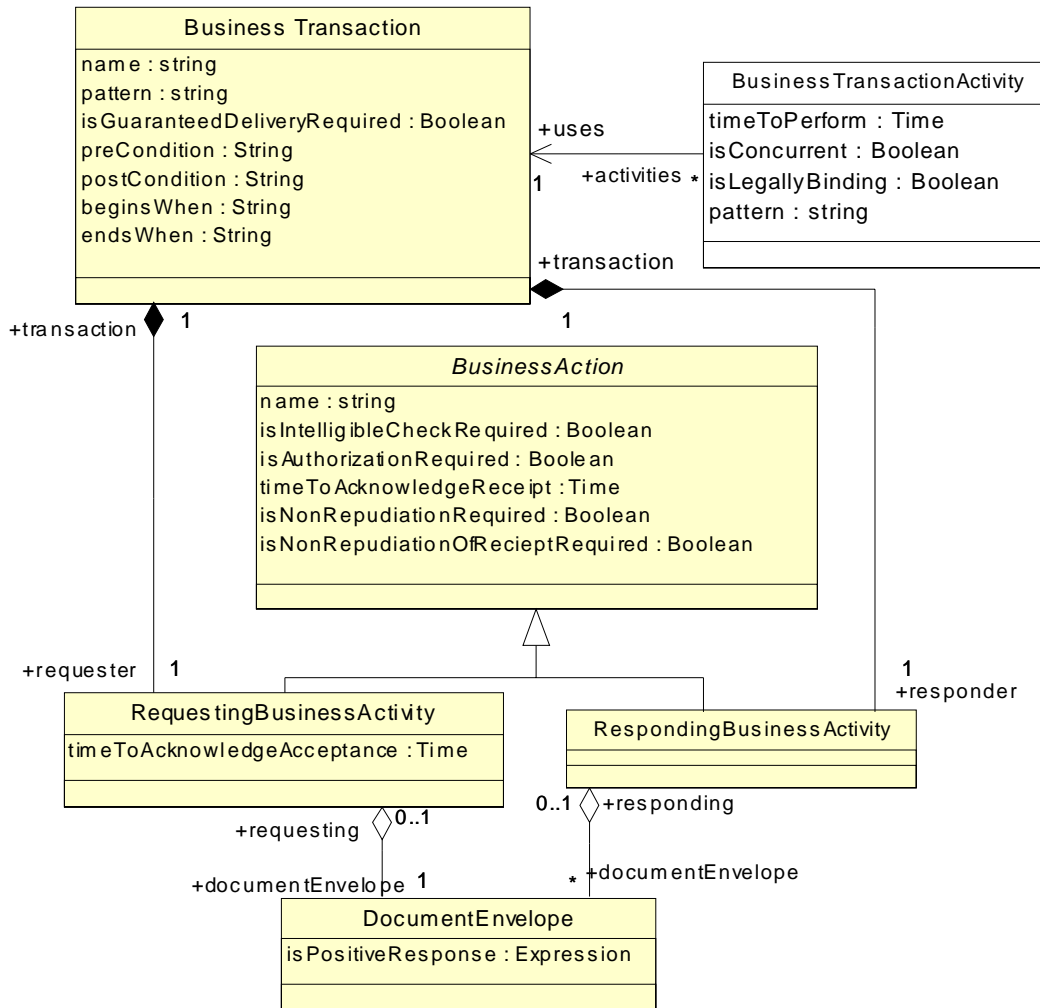
667

668

669 **6.4.1 Specify a Business Transaction and its Business Document** 670 **Flow**

671

672 Figure 6 illustrates a business transaction.



673

674

675

Figure 6. UML Diagram of a Business Transaction

676

677 6.4.1.1 Key Semantics of a Business Transaction

678

679

680

A Business Transaction is the atomic unit of work in a trading arrangement between two business partners.

681

682

683

684

685

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 be additionally supported by one or more Business Signals that govern the use and meaning of acknowledgements and related matters in the transaction.

686 Implicitly there is a requesting role performing the Requesting Business
687 Activity and a responding role performing the Responding Business
688 Activity. These roles become explicit when the transaction is used within
689 a Business Transaction Activity within a Binary Collaboration.

690 There is always a Request document flow.

691 Whether a Response document flow is required is part of the definition of
692 the Business Transaction. Some Business Transactions need this type of
693 request and response, typically for the formation of a contract or
694 agreement. Other Business Transactions are more like notifications, and
695 have only a Request document flow.

696 An abstract superclass, Business Action, is the holder of attributes that
697 are common to both Requesting Business Activity and Responding
698 Business Activity.

699 6.4.1.2 Sample syntax

700 Here is a simple notification transaction with just one document flow:

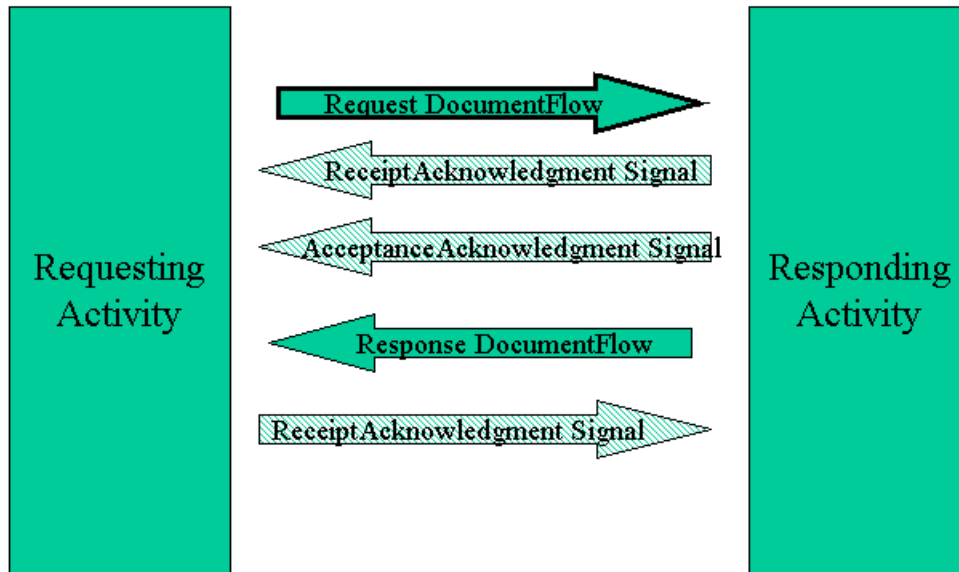
```
701 <BusinessTransaction name="Notify of advanceshipment">  
702     <RequestingBusinessActivity name="">  
703         <DocumentEnvelope isPositiveResponse="true"  
704             BusinessDocument name="ASN"/>  
705     </RequestingBusinessActivity>  
706     <RespondingBusinessActivity name="">  
707     </RespondingBusinessActivity>  
708 </BusinessTransaction>  
709
```

710 Associated with each document flow can be one or more business signals
711 acknowledging the document flow. These acknowledgment signals are
712 not modeled explicitly but parameters associated with the transaction
713 specify whether the signals are required or not.

714 The possible Document Flows and business signals within a Business
715 Transaction are shown in Figure 7.

716

717



718
719 Figure 7: Possible document flows and signals and their sequence

720

721

722

These acknowledgment signals (a.k.a. Business Signals) are application level documents that 'signal' the current state of the business transaction.

723

724

725

726

727

728

729

Whether a receiptAcknowledgement and/or acceptanceAcknowledgement signal are required is part of the pattern specified for the Business Transaction. These business signals have specific business purposes, relating to the processing and management of documents and document envelopes *prior* to evaluation of their business terms, and are separate from lower protocol and transport signals.

730

731

732

733

734

735

736

737

738

739

The Receipt acknowledgement business signal, if used, signals that a message has been properly received. The property *isIntelligibleCheckRequired* allows partners to agree that a message should be confirmed by a Receipt acknowledgement only if it also is legible. Legible means that it has been passed a structure/ schema validity check. Both the proper receipt and, if evaluated, the legibility of a message are reviewed (and if present acknowledged) *prior* to the application of any business rules or evaluation of the terms or guard expressions in the message's business documents or document envelope,

740 The Acceptance Acknowledgement business signal, if used, signals that
 741 the message received has been accepted for business processing. This
 742 is the case if the contents of the message's business documents and
 743 document envelope have passed a business rule validity check.

744 Failure to send either signal, when required (by specifying a timeout value
 745 in `timeToAcknowledgeReceipt` or `timeToAcknowledgeAcceptance`), will
 746 result in the transaction being null and void, and therefore will prevent any
 747 "success" end state that would have depended on receipt of a business
 748 document satisfying the associated `timeToPerform`.

749

750 6.4.1.3 Sample syntax

751 Here is a slightly more complex transaction with two document flows and
 752 three business signals.

753 The request requires both receipt and acceptance acknowledgement, the
 754 response requires only receipt acknowledgement. "P2D" is a W3C
 755 Schema syntax adopted from the ISO 8601 standard and means
 756 Period=2 Days. P3D means Period=3 Days, P5D means Period=5 Days.
 757 These periods are all measured from original sending of request.

```
758 <BusinessTransaction name="Create Order">
759     <RequestingBusinessActivity name=" "
760         isNonRepudiationRequired="true"
761         timeToAcknowledgeReceipt="P2D"
762         timeToAcknowledgeAcceptance="P3D">
763         <DocumentEnvelope isPositiveResponse="true"
764             BusinessDocument="Purchase Order"/>
765     </RequestingBusinessActivity>
766     <RespondingBusinessActivity name=" "
767         isNonRepudiationRequired="true"
768         timeToAcknowledgeReceipt="P5D">
769         <DocumentEnvelope isPositiveResponse="true"
770             BusinessDocument="PO
771             Acknowledgement"/>
772     </RespondingBusinessActivity>
773 </BusinessTransaction>
```

776 6.4.1.4 Specifying Business Document flows

777 Request document flows and response document flows contain Business
 778 Documents that pertain to the Business Transaction. The model for this is
 779 shown in Figure 8. Business Documents have varying structures.
 780 Business signals, however always have the same structure, defined once
 781 and for all as part of the ebXML *Business Process Specification Schema*.

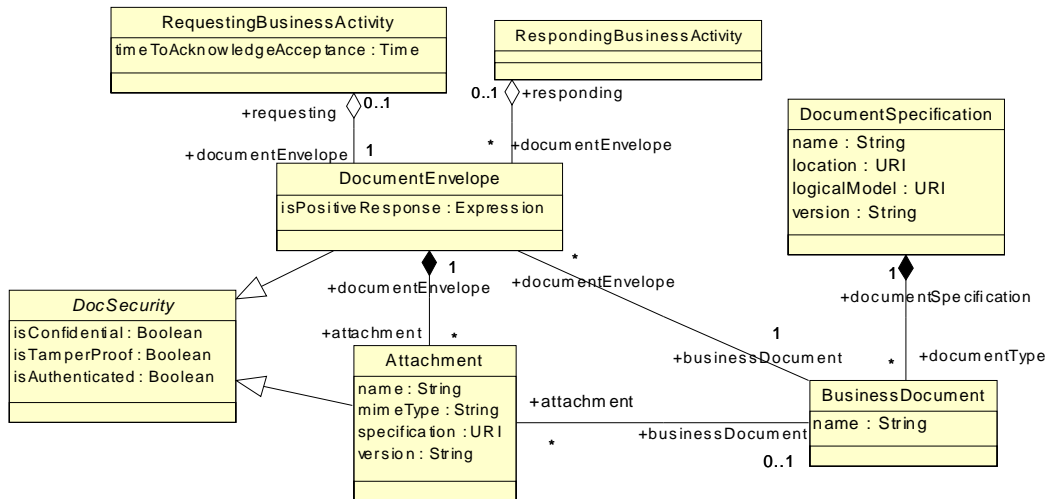
784

785

786

787

788



789

790

Figure 8: UML Diagram of document flow

791

792

793

794

795

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 Business Activity and one Responding Business Activity to model the flow.

796

797

798

799

800

801

802

803

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. For example, the Response Document Envelopes for a purchase order transaction might be named PurchaseOrderAcceptance, PurchaseOrderDenial, and PartialPurchaseOrderAcceptance. In the actual execution of the purchase order transaction, however, only one of the defined possible responses will be sent.

804

805

806

807

808

The Document Envelope represents the flow of documents between the activities. Each Document Envelope carries exactly one primary Business Document. A Business Document is defined in a DocumentSpecification. This may be an ebXML DocumentSpecification, or a DocumentSpecification supplied by an outside source.

809

810

811

812

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.

813 6.4.1.5 Sample syntax

814 This example shows a business transaction with one request and two
 815 possible responses, a success and a failure. The request has an
 816 attachment. All the documents are defined in a named
 817 DocumentSpecification, and the Business Documents are fully qualified
 818 with the schema name.

819

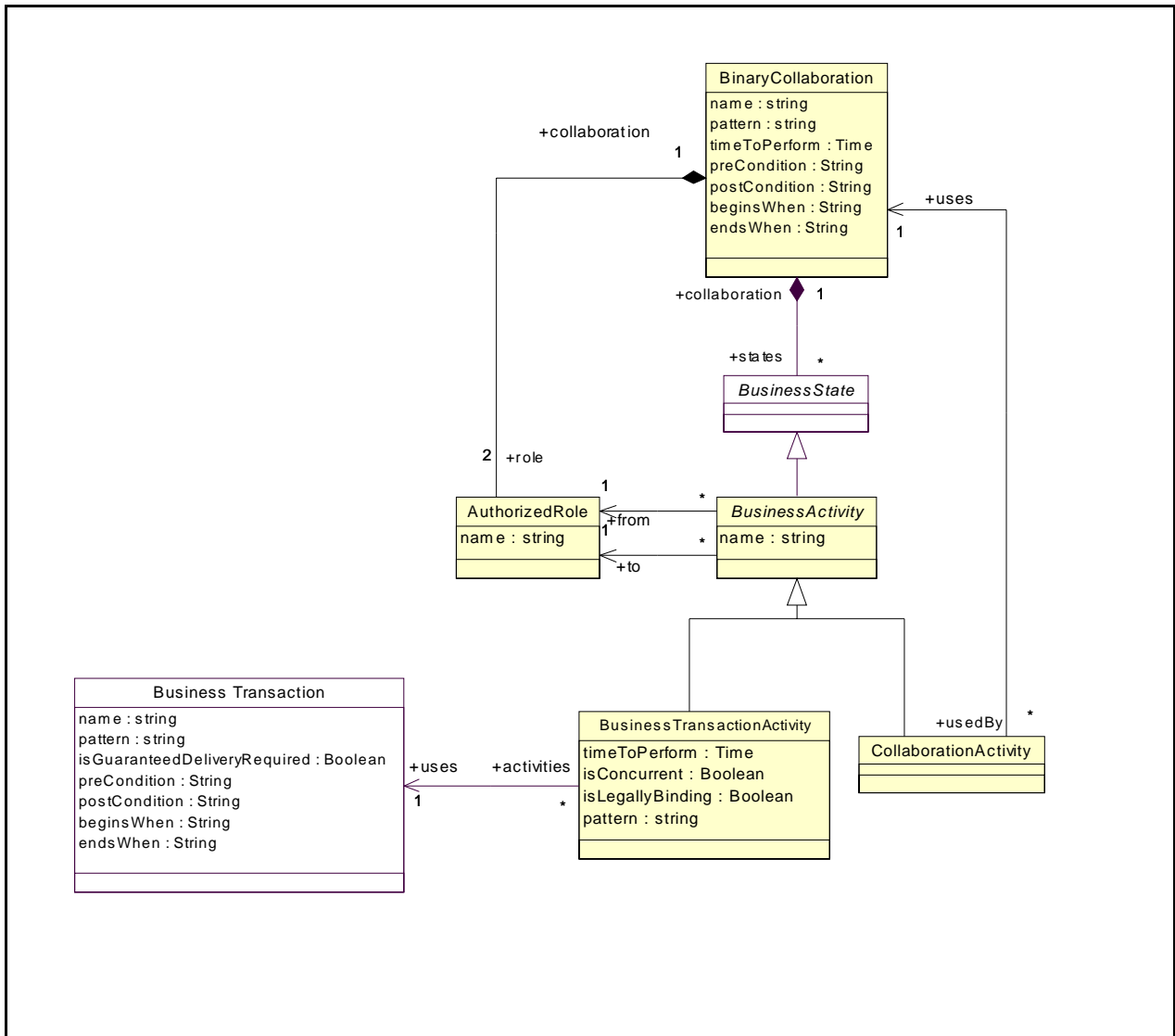
```

820 <DocumentSpecification name="ebXML1.0" location="someplace"
821     logicalModel="someplaceAlso">
822     <BusinessDocument name=" Purchase Order "/>
823     <BusinessDocument name=" PO Acknowledgement "/>
824     <BusinessDocument name=" PO Rejection "/>
825     <BusinessDocument name="Delivery Instructions"/>
826 </DocumentSpecification>
827
828 <BusinessTransaction name="Create Order">
829     <RequestingBusinessActivity name=""
830     <DocumentEnvelope isPositiveResponse="true"
831     BusinessDocument="ebXML1.0/PO Acknowledgement">
832     <Attachment
833     name="DeliveryNotes"
834     mimeType="XML"
835     BusinessDocument=
836     "ebXML1.0/Delivery Instructions"
837     specification=""
838     isConfidential="true"
839     isTamperProof="true"
840     isAuthenticated="true">
841     </Attachment>
842     </DocumentEnvelope>
843 </RequestingBusinessActivity>
844 <RespondingBusinessActivity name=""
845     <DocumentEnvelope isPositiveResponse="true"
846     BusinessDocument="ebXML1.0/PO
847 Acknowledgement"/>
848     </DocumentEnvelope>
849     <DocumentEnvelope isPositiveResponse="false"
850     BusinessDocument=" ebXML1.0/PO Rejection"/>
851     </DocumentEnvelope>
852 </RespondingBusinessActivity>
853 </BusinessTransaction>
854

```

855 6.4.2 Specify a Binary Collaboration

856 Figure 9 illustrates a binary collaboration.



857

858

859

Figure 9: UML Diagram of a Binary Collaboration

860

861

862 6.4.2.1 Key Semantics of a Binary Collaboration

863 A Binary Collaboration is always between two roles. These two roles are
 864 called Authorized Roles, because they represent the actors that are
 865 authorized to participate in the collaboration.

866 A Binary Collaboration consists of one or more Business Activities. These
 867 Business Activities are always conducted **between** the two Authorized
 868 Roles of the Binary Collaboration. For each activity one of two roles is
 869 assigned to be the initiator (from) and the other to be the responder (to).

870 A Business Activity can be either a Business Transaction Activity or a
871 Collaboration Activity.

872 A Business Transaction Activity is the performance of a Business
873 Transaction. Business Transactions are re-useable relative to Business
874 Transaction Activity. The same Business Transaction can be performed
875 by multiple Business Transaction Activities in different Binary
876 Collaborations, or even by multiple Business Transaction Activities in the
877 same Binary Collaboration.

878 A Collaboration Activity is the performance of a Binary Collaboration,
879 possibly within another Binary Collaboration. Binary Collaborations are re-
880 useable relative to Collaboration Activity. The same Binary Collaboration
881 can be performed by multiple Collaboration Activities in different Binary
882 Collaborations, or even by multiple Collaboration Activities in the same
883 Binary Collaboration.

884 When performing a Binary Collaboration within a Binary Collaboration
885 there is an implicit relationship between the roles at the two levels.
886 Assume that Binary Collaboration X is performing Binary Collaboration Y
887 through Collaboration Activity Q. Binary Collaboration X has Authorized
888 roles Customer and Vendor. In Collaboration Activity Q we assign
889 Customer to be the initiator, and Vendor to be the responder. Binary
890 Collaboration X has Authorized roles Buyer and Seller and a Business
891 Transaction Activity where Buyer is the initiator and Seller the responder.
892 We have now established a role relationship between the roles Customer
893 and Buyer because they are both initiators in activities in the related
894 performing and performed Binary Collaborations.

895 Since a Business Transaction is atomic in nature, the performing of a
896 single Business Transaction through a Business Transaction Activity is
897 also atomic in nature. If the desired semantic is not atomic, then the task
898 should be split over multiple transactions. For instance if it is desired to
899 model several partial acceptances of a request, then the request should
900 be modeled as one transaction within a binary collaboration and the
901 partial acceptance(s) as separate transactions.

902 The CPA/CPP Specification requires that parties agree upon a
903 Collaboration Protocol Agreement (CPA) in order to transact business. A
904 CPA associates itself with a specific Binary Collaboration. Thus, all
905 Business Transactions performed between two parties should be
906 referenced through Business Transaction Activities contained within a
907 Binary Collaboration.

908

909 6.4.2.2 Sample syntax

910

911 Here is a simple Binary Collaboration using one of the Business
912 Transactions defined above:

913

```

914     <BinaryCollaboration name="Firm Order"
915     timeToPerform="P2D">
916         <Documentation>
917             timeToPerform =
918             Period: 2 days from start of transaction
919         </Documentation>
920         <AuthorizedRole name="buyer"/>
921         <AuthorizedRole name="seller"/>
922         <BusinessTransactionActivity name="Create Order"
923             businessTransaction="Create Order"
924             fromAuthorizedRole="buyer"
925             toAuthorizedRole="seller"/>
926     </BinaryCollaboration>

```

927

928 Here is a slightly more complex Binary Collaboration re-using the same
 929 Business Transaction as the previous Binary Collaboration, and adding the
 930 use of another of the Business Transactions defined above.:

931

```

932     <BinaryCollaboration name="Product Fulfillment"
933     timeToPerform="P5D">
934         <Documentation>
935             timeToPerform =
936             Period: 5 days from start of transaction
937         </Documentation>
938         <AuthorizedRole name="buyer"/>
939         <AuthorizedRole name="seller"/>
940         <BusinessTransactionActivity name="Create Order"
941             businessTransaction="Create Order"
942             fromAuthorizedRole="buyer"
943             toAuthorizedRole="seller"
944             isLegallyBinding="true" />
945         <BusinessTransactionActivity
946             name="Notify shipment"
947             businessTransaction="Notify of advance
948             shipment"
949             fromAuthorizedRole="buyer"
950             toAuthorizedRole="seller"/>
951     </BinaryCollaboration>

```

952

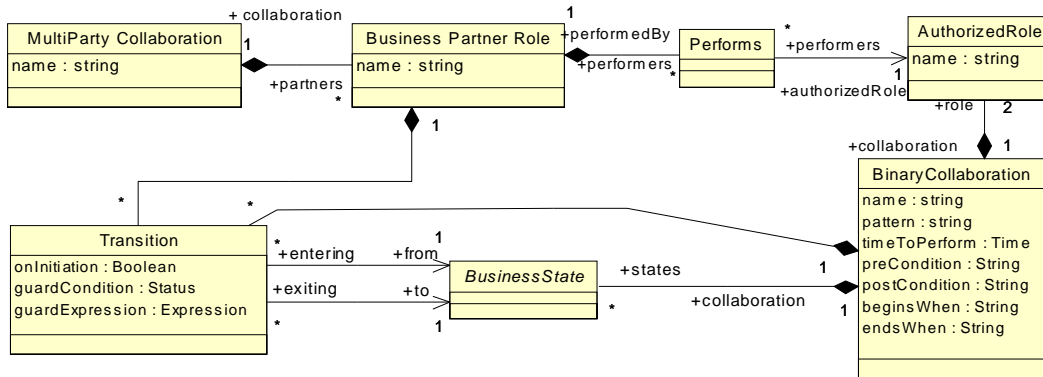
953

954

955 6.4.3 Specify a MultiParty Collaboration

956 Figure 10 illustrates a multiparty collaboration

957



958

959

960

Figure 10: UML Diagram of a MultiParty Collaboration

961

962

963 6.4.3.1 Key Semantics of a MultiParty Collaboration

964

A MultiParty Collaboration is a synthesis of Binary Collaborations.

965

A MultiParty Collaboration consists of a number of Business Partner Roles.

966

967

Each Business Partner Role performs one Authorized Role in one of the binary collaborations, or perhaps one Authorized Role in each of several binary collaborations. This is modeled by use of the Performs element.

968

969

970

This 'Performs' linkage between a Business Partner Role and an Authorized Role is the synthesis of Binary Collaborations into MultiParty Collaborations. Implicitly the MultiParty Collaboration consists of all the Binary Collaborations in which its Business Partner Roles play Authorized Roles.

971

972

973

974

975

Each binary pair of trading partners will be subject to one or more distinct CPAs.

976

977

Within a MultiParty Collaboration, you may choreograph transitions between Business Transaction Activities in different Binary Collaborations, as described below.

978

979

980

981 6.4.3.2 Sample syntax

982

Here is a simple MultiParty Collaboration using the Binary Collaborations defined above.

983

984

```
<MultiPartyCollaboration name="DropShip">
```

985

```
<BusinessPartnerRole name="Customer">
```

986

```
<Performs
```

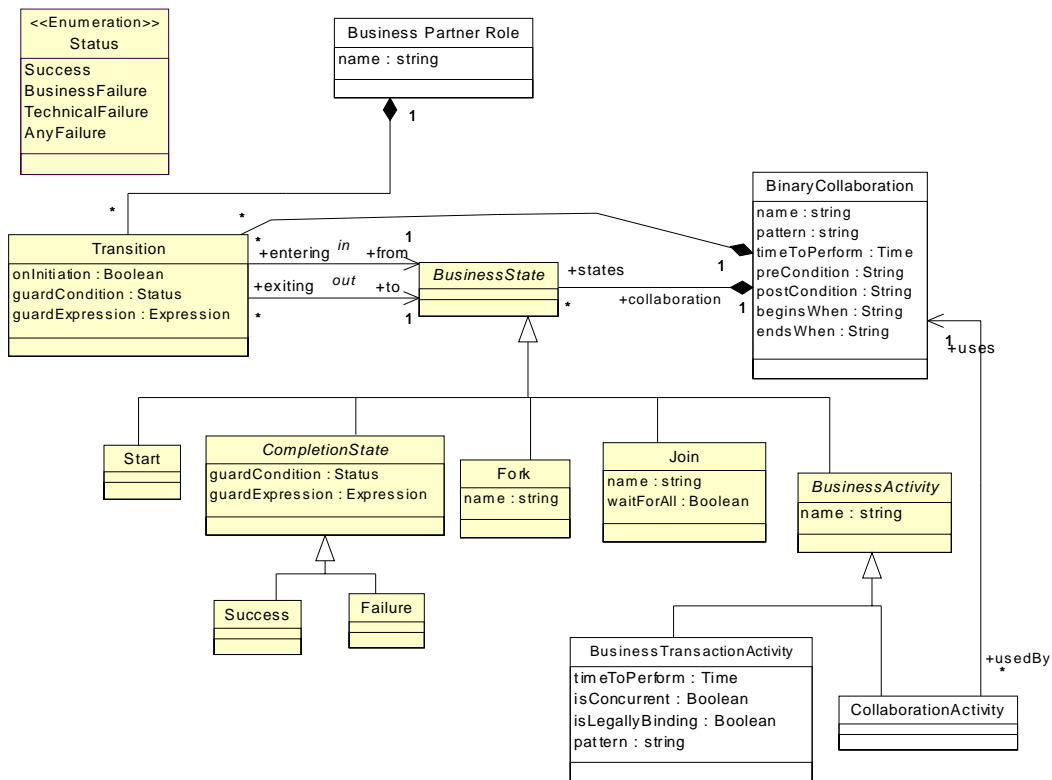
987

```
authorizedRole=
```

```
988         `//binaryCollaboration[@name="Firm Order"]
989         /AuthorizedRole[@name="buyer"]' />
990     </BusinessPartnerRole>
991     <BusinessPartnerRole name="Retailer">
992         <Performs
993         authorizedRole=
994         `//binaryCollaboration[@name="Firm Order"]
995         /AuthorizedRole[@name="seller"]' />
996         <Performs
997         `//binaryCollaboration[@name=" Product
998         Fulfillment" /AuthorizedRole[@name="buyer"]' />
999     </BusinessPartnerRole>
1000     <BusinessPartnerRole name="DropShip Vendor">
1001         <Performs
1002         `//binaryCollaboration[@name=" Product
1003         Fulfillment"
1004         /AuthorizedRole[@name="seller"]' />
1005     </BusinessPartnerRole>
1006 </MultiPartyCollaboration>
1007
```

1008 6.4.4 Specify a Choreography

1009 Figure 11 illustrates a choreography.



1010

1011

1012

Figure 11: UML Diagram of a Choreography

1013

1014 6.4.4.1 Key Semantics of a Choreography

1015

1016 A Choreography is an ordering and sequencing of Business Activities
 1017 within a Binary Collaboration.

1018

1019

The choreography is specified in terms of Business States, and
 transitions between those Business States.

1020

1021

1022

1023

1024

1025

A Business Activity is an abstract kind of Business State. Its two subtypes
 Business Transaction Activity and Collaboration Activity are concrete
 Business States. The purpose of a Choreography is to order and
 sequence Business Transaction Activity and/or Collaboration Activity
 within a Binary Collaboration, or across Binary Collaborations within a
 Multiparty Collaboration.

1026

1027

1028

There are a number of auxiliary kinds of Business States that facilitate the
 choreographing of Business Activities. These include a Start state, a
 Completion state (which comes in a Success and Failure flavor), a Fork

1029 state and a Synchronization state. These are all equivalent to
 1030 diagramming artifacts on a UML activity chart.

1031 Transitions are between Business States. Transitions can be gated by
 1032 Guards. Guards can refer to the status of the Document Envelope that
 1033 caused the transition, the type of Document sent, the content of the
 1034 document, or postconditions on the prior state.

1035 A Transition can also be used to create nested
 1036 BusinessTransactionActivities. A nested BusinessTransactionActivity is
 1037 one where a first transition happens after the receipt of the request in the
 1038 first transaction, and then the entire second transaction is performed
 1039 before returning to the first transaction to send the response back to the
 1040 original requestor. The flag 'onInitiation' in Transition is used for this
 1041 purpose. Nested BusinessTransactionActivity are typically within a
 1042 multiparty collaboration. In essence an Authorized Role in one Binary
 1043 Collaboration receives a request, then turns around and becomes the
 1044 requestor in an other Binary Collaboration before coming back and
 1045 sending the response in the first Binary Collaboration.

1046 isConcurrent is a parameter that governs the flow of transactions. Unlike
 1047 the security and timing parameters it does not govern the internal flow of
 1048 a transaction, rather it determines whether multiple instances of that
 1049 transaction type can be 'open' at the same time as part of the same
 1050 business transaction activity. IsConcurrent is the parameter that governs
 1051 this. It is at the business transaction activity level.

1052

1053 6.4.4.2 Sample syntax

1054

1055 Here is the same Binary Collaboration as used before, with choreography
 1056 added at the end. There is a transition between the two, a start and two
 1057 possible outcomes of this collaboration, success and failure:

```

1058 <BinaryCollaboration name="Product Fulfillment"
1059 timeToPerform="P5D">
1060     <Documentation>
1061         timeToPerform =
1062         Period: 5 days from start of transaction
1063     </Documentation>
1064     <AuthorizedRole name="buyer"/>
1065     <AuthorizedRole name="seller"/>
1066     <BusinessTransactionActivity name="Create Order"
1067         businessTransaction="Create Order"
1068         fromAuthorizedRole="buyer"
1069         toAuthorizedRole="seller"/>
1070     <BusinessTransactionActivity
1071         name="Notify shipment"
1072         businessTransaction="Notify of advance
1073         shipment"
1074         fromAuthorizedRole="buyer"
  
```

```

1075         toAuthorizedRole="seller"/>
1076     <Start toBusinessState="Create Order"/>
1077     <Transition
1078         fromBusinessState="Create Order"
1079         toBusinessState="Notify shipment"/>
1080     <Success fromBusinessState="Notify shipment"
1081         guardCondition="Success"/>
1082     <Failure fromBusinessState="Notify shipment"
1083         guardCondition="BusinessFailure"/>
1084 </BinaryCollaboration>
1085

```

1086 Here is the same Multipart Collaboration as defined before, but with a simple
1087 choreography (transition) across two Binary Collaborations.

```

1088
1089 <MultiPartyCollaboration name="DropShip">
1090     <BusinessPartnerRole name="Customer">
1091         <Performs
1092             authorizedRole=
1093             `//binaryCollaboration[@name="Firm Order"]
1094             /AuthorizedRole[@name="buyer"]' />
1095     </BusinessPartnerRole>
1096     <BusinessPartnerRole name="Retailer">
1097         <Performs
1098             authorizedRole=
1099             `//binaryCollaboration[@name="Firm Order"]
1100             /AuthorizedRole[@name="seller"]' />
1101         <Performs
1102             `//binaryCollaboration[@name=" Product
1103             Fulfillment" /AuthorizedRole[@name="buyer"]' />
1104         <Transition
1105             fromBinaryCollaboration="Firm Order"
1106             fromBusinessState=
1107             `//binaryCollaboration[@name="Firm Order"]
1108             /[@name="Create Order"]'
1109             toBusinessState=
1110             `//binaryCollaboration[@name="Product
1111             Fulfillment"]
1112             /[@name="Create Order"]'
1113         />
1114     </BusinessPartnerRole>
1115     <BusinessPartnerRole name="DropShip Vendor">
1116         <Performs
1117             `//binaryCollaboration[@name=" Product
1118             Fulfillment"
1119             /AuthorizedRole[@name="seller"]' />
1120     </BusinessPartnerRole>
1121 </MultiPartyCollaboration>
1122
1123

```

1124 6.4.5 The whole model

1125

1126

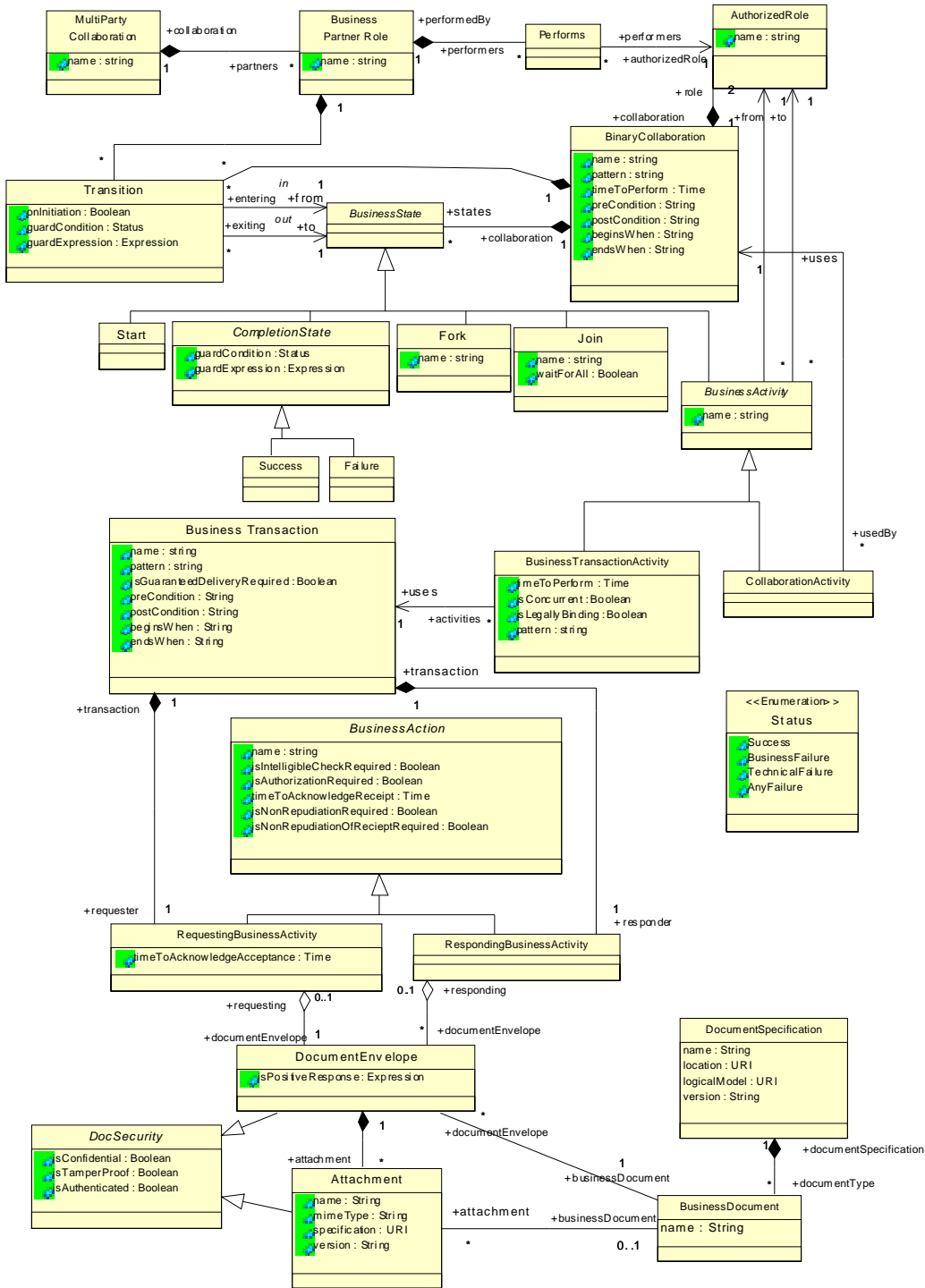
1127

1128

Figure 12 shows the above semantics collectively as a UML class diagram. This diagram contains the whole UML version of the ebXML *Business Process Specification Schema*

1129
1130

Figure 12: Overall ebXML Business Process Specification Schema as UML class diagram



1131

1132

1133 **6.5 Core Business Transaction Semantics**

1134 The ebXML concept of a business transaction and the semantics behind it are
1135 central to predictable, enforceable commerce. It is expected that any Business
1136 Service Interface (BSI) will be capable of managing a transaction according to
1137 these semantics.

1138 The ebXML Business Transaction semantics allows you to specify electronic
1139 commerce transactions that provide

- 1140 • Interaction Predictability, i.e. have clear roles, clear transaction scope,
1141 clear time bounds, clear business information semantics, clear
1142 determination of success or failure.
- 1143 • Ability to create Legally Binding Contracts, i.e. the ability to specify that
1144 Business Transactions may be agreed to bind the parties.
- 1145 • Nonrepudiation, i.e. may specify the keeping of artifacts to aid in legal
1146 enforceability.
- 1147 • Authorization Security, i.e. may be specified to require athorization of
1148 parties performing roles.
- 1149 • Document Security, i.e. may be specified to be authorized, authenticated,
1150 confidential, tamperproof.
- 1151 • Reliability, i.e. the ability to specify reliable delivery of Business
1152 Documents and signals.
- 1153 • Run time Business Transaction Semantics, i.e. the rules and
1154 configuration parameters required for Business Service Interface software
1155 to predictably and deterministically execute ebXML Business
1156 Transactions.

1157 Each of the above characteristics of ebXML Business Transaction semantics is
1158 discussed in detail below

1159 **6.5.1 Interaction Predictability**

1160

1161

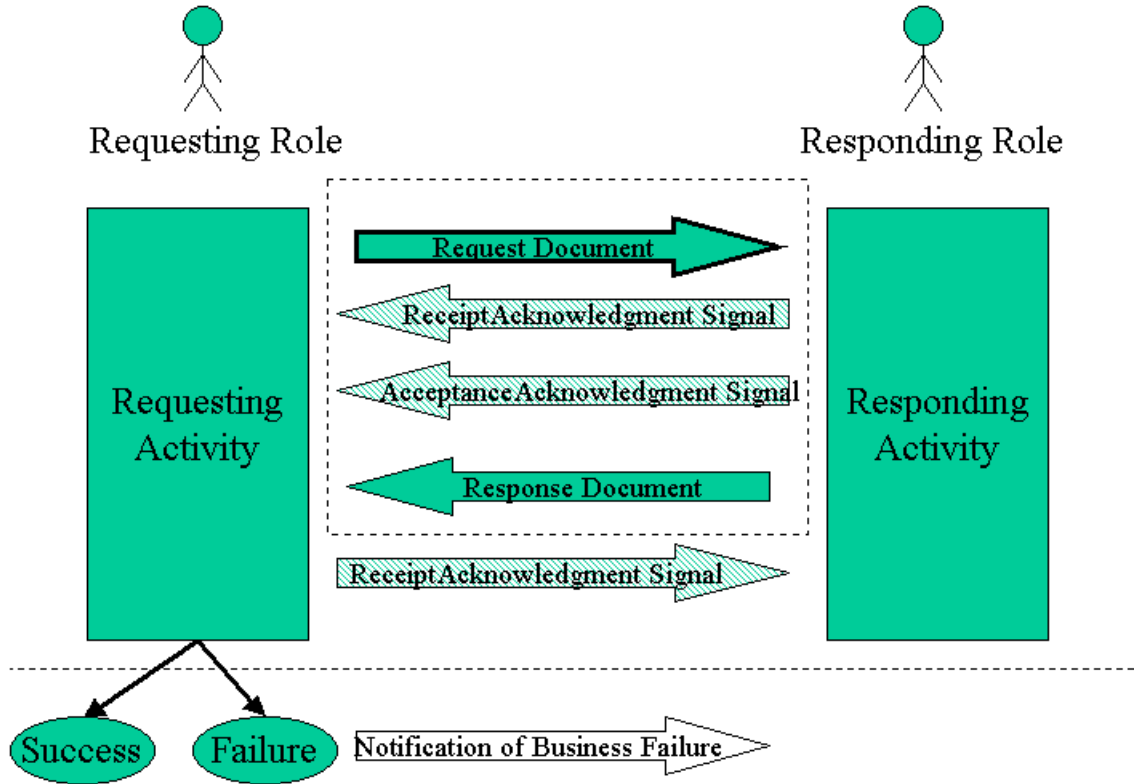
1162

1163

1164

1165

All Business Transactions follow a very precisely prescribed flow, or a precisely defined subset there-of. The following is an overall illustration of this flow. It can be thought of as the state machine across the two business partners. The N090R9.1 chapter on the UMM metamodel has a detail state chart for each of the business partners.



1166

1167

Figure 13: Schematic of core Business Transaction semantics.

1168

1169

In the ebXML model the business transaction always has the following semantics.

1170

1171

1. The Business Transaction is a unit of work. All of the interactions in a business transaction must succeed or the transaction must be rolled back to a defined state before the transaction was initiated.

1172

1173

1174

2. A Business Transaction is conducted between two business partners playing opposite roles in the transaction. These roles are always the Requesting Role and the Responding Role.

1175

1176

1177

3. A Business Transaction definition specifies exactly when the Requesting Activity is in control, when the Responding Activity is in control, and when control transitions from one to the other. In all Business Transactions control starts at the Requesting Activity, then transitions to the Responding Activity, and then returns to the Requesting Activity.

1178

1179

1180

1181

1182

4. A Business Transaction always starts with a request sent out by the requesting activity.

1183

1184

5. The request serves to transition control to the responding role.

- 1185 6. After the receipt of the Request document flow, the responding activity
1186 may send a receiptAcknowledgement signal and/or an
1187 acceptanceAcknowledgement signal to the requesting role.
- 1188 7. The responding role then enters a responding activity. During or upon
1189 completion of the responding activity zero or one response is sent.
- 1190 8. Control will be returned back to the requesting activity if either a
1191 receiptAcknowledgement and/or acceptanceAcknowledgement and/or a
1192 response is specified as required. A receiptAcknowledgement (if
1193 required) must always occur before an acceptanceAcknowledgement (if
1194 required), and an acceptanceAcknowledgement must always occur
1195 before a response (if required). Control is returned to the requesting
1196 activity based on the last required of these three (if any). If none required,
1197 control stays with the responding activity.
- 1198 9. All business transactions succeed or fail. Success or failure depends on:
1199 a. The receipt or non-receipt of the request, the response and/or
1200 business signals
1201 b. The occurrence of time-outs
1202 c. The occurrence of a business exception
1203 d. The occurrence of a control exception
1204 e. The interpretation of the received response and guard
1205 expressions on transitions to success or failure
- 1206 10. The determination of Business Transaction success or failure is
1207 established by the requesting party based on the above success or
1208 failure factors. Once success or failure is thus established, the Business
1209 Transaction is considered closed with respect to both parties.
- 1210 11. Upon receipt of a response the requesting activity may send a
1211 receiptAcknowledgement signal back to the responding role. This is
1212 merely a signal and does not pass control back to the responding activity,
1213 nor does it alter the successful or failed completion of the Business
1214 Transaction that was based on the receipt of the Response.
- 1215 12. Upon identifying a time-out exception in the processing of a Business
1216 Transaction, and closing the transaction accordingly, the requesting party
1217 may send a notification of failure to the responding party. This is
1218 considered a new Business Transaction and does not alter the already
1219 established conclusion of the Business Transaction.

1220

1221 6.5.1.1 Transaction Interaction Patterns

1222

1223

1224

1225

1226

1227

The business transaction specification will specify whether a requesting document requires a responding substantive document in order to achieve a "success" end state. In addition, the transaction may specify a proper nonzero time duration for timeToPerform, imposing a deadline for the substantive response.

1228 Furthermore, the specification of a business transaction may
1229 indicate, for the request whether receiptAcknowledgement and/or
1230 acceptanceAcknowledgement are required, and for the response
1231 whether receiptAcknowledgement is required.

1232 The way to specify that a receiptAcknowledgement is required is
1233 to set the parameter timeToAcknowledgeReceipt to any proper
1234 time duration other than zero. If this parameter has been set to a
1235 proper nonzero time duration, optionally either or both of the
1236 isIntelligibleCheckRequired and
1237 isNonrepudiationOfReceiptRequired parameters may also be set
1238 to 'Yes'.

1239 The way to specify that a acceptanceAcknowledgement is
1240 required is to set the parameter timeToAcknowledgeAcceptance
1241 to any proper time duration other than zero.

1242 So these two acknowledgement related parameters double as
1243 Boolean flags for whether the signal is required as part of the
1244 transaction, and as values for time-out of the transaction if the
1245 signal is not received.

1246 The specification of a business transaction may require each one
1247 of these signals independently of whether the other is required. If
1248 one is not required, it is actually not allowed. Therefore there is a
1249 finite set of combinations. The UMM supplies an illustrative set of
1250 patterns representing those combinations, for potential re-use.

1251

1252 6.5.2 Creating legally binding contracts

1253

1254 Trading partners may wish to indicate that a Business Transaction
1255 performed as part of an ebXML arrangement is, or is not, intended to be
1256 binding. A declaration of intent to be bound is a key element in
1257 establishing the legal equivalence of an electronic message to an
1258 enforceable signed physical writing. Parties may create explicit evidence
1259 of that intent by (1) adopting this standard and (2) manipulating the
1260 parameter ("isLegallyBinding") designated by the standard to indicate that
1261 intent.

1262

1263 In some early electronic applications, trading partners have simply used
1264 the presence, or absence, of an electronic signature (such as under the
1265 XML-DSIG standard) to indicate that intent. However, documents which
1266 rely solely on the presence of a signature may or may not be correctly
1267 interpreted, if there is semantic content indicating that a so-called
1268 contract is a draft, or nonbinding, or the like.

1269 In ebXML, the presence or absence of an electronic signature cannot
1270 indicate by itself determine legally binding assent, because XML-DSIG
1271 signatures are reserved for other uses as an assurance of sender identity
1272 and message integrity.

1273

1274 isLegallyBinding is a parameter at the BusinessTransactionActivity level,
1275 which means that the performing of a BusinessTransaction within a
1276 Binary Collaboration is either specified as legally binding or not.
1277
1278 When operating under this standard, parties form binding agreements by
1279 exchanging binding messages that agree to terms (e.g., offer and
1280 acceptance).
1281 The "isLegallyBinding" parameter is Boolean, and its default value is
1282 "true." Under this standard, the exclusive manner for indicating that a
1283 Business Activity is not intended to be binding is to include a "false"
1284 value for the "isLegallyBinding" parameter for the transaction activity. As
1285 in EDI, the ebXML standard assumes that Business Transactions are
1286 intended by the trading parties to be binding unless otherwise indicated.
1287
1288 As a non-normative matter, parties may wish to conduct nonbinding
1289 transactions for a variety of reasons, including testing, and the exchange
1290 of proposed offers and counteroffers on a non-committal basis so as to
1291 discover a possible agreed set of terms. When using tangible signed
1292 documents, parties often do so by withholding a manual signature, or
1293 using a "DRAFT" stamp. In ebXML, trading partners may indicate that
1294 result by use of the "isLegallyBinding" parameter. See the illustrative
1295 Simple Negotiation Pattern set forth in the ebXML E-Commerce and
1296 Simple Negotiation Patterns.

1297 6.5.3 Non-Repudiation

1298
1299 Trading partners may wish to conduct legally enforceable
1300 business transactions over ebXML. A party may elect to use non-
1301 repudiation protocols in order to generate documentation that
1302 would assist in the enforcement of the contractual obligation in
1303 court, in the case that the counterparty later attempts to repudiate
1304 its ebXML Business Documents and messages.
1305
1306 Repudiation generally refers to the ability of a trading partner to
1307 argue at a later time, based on the persistent artifacts of a
1308 transaction, that it did not agree to the transaction. That argument
1309 might be based on assertions that a replying document was not
1310 sent, or was not sent by the proper party, or was incorrectly
1311 interpreted (under the applicable standard or the trading partners'
business rules) as forming agreement.
1312
1313 There are two kinds of non-repudiation protocol available under
1314 this document. Each protocol provides the user with some degree
1315 of additional evidentiary assurance by creating or requesting
1316 additional artifacts that would assist in a later dispute over
1317 repudiation issues. Neither is a dispositive absolute assurance.
1318 As in the paper world, trading partners are always free to invent
1319 colorful new arguments than an apparently-enforceable statement
1320 should be ignored. These parameters simply offer some
opportunities to make that more difficult.

1321 One imposes a duty on each party to save copies of all Business
 1322 Documents and Document Envelopes comprising the transaction,
 1323 each on their own side, i.e., requestor saves his request,
 1324 responder saves his response. This is the
 1325 isNonRepudiationRequired parameter in the requesting or
 1326 responding activity. It is logically equivalent to a request that the
 1327 other trading partner maintain an audit trail. However, failure to
 1328 comply with that request is not necessarily computationally
 1329 detectable at run time, nor would it affect this schema's
 1330 determination of a "success" or "failure" end state..

1331 The other requires the responder to send a signed copy of the
 1332 receipt, which the requestor then saves. This is the
 1333 isNonRepudiationOfReceiptRequired parameter in the requesting
 1334 business activity.

1335 NonRepudiationOfReceipt is tied to the ReceiptAcknowledgement,
 1336 in that it requires the latter to be digitally signed. So
 1337 NonRepudiationOfReceipt is meaningless if
 1338 ReceiptAcknowledgement is not required. Failure to comply with
 1339 NonRepudiation of Receipt would be computationally detectable
 1340 at run time, and would affect this schema's determination of a
 1341 "failure" end state. If a timeToAcknowledgeReceipt is imposed on
 1342 a requesting message, and NonRepudiationOfReceipt is true,
 1343 only a digitally signed receipt will satisfy the imposed timeout
 1344 deadline. Thus, a failure to send a *signed* receipt within
 1345 timeToAcknowledgeReceipt, would make the transaction null and
 1346 void.

Parameter	BSI requirement
isNonRepudiationRequired	Must save audit trail of messages it sends
isNonRepudiationOfReceiptRequired	Must digitally sign receiptAcknowledgements

1347

1348 6.5.4 Authorization security

1349

1350 Each request or response may be sent by a variety of individuals,
 1351 representatives or automated systems associated with a business
 1352 partner. There may be cases where trading partners have more
 1353 than one ebXML-capable business service interface, representing
 1354 different levels of authority. In such a case, the parties may
 1355 establish rules regarding which interfaces or authors may be
 1356 confidently relied upon as speaking for the enterprise.

1357 In order to invoke those rules, a party may specify
 1358 IsAuthorizationRequired on a requesting or and responding
 1359 activity accordingly, with the result that [the activity] will only be

1360 processed as valid if the party interpreting it successfully matches
 1361 the stated identity of the activity's [Authorized Role] to a list of
 1362 allowed values previously supplied by that party.

Parameter	BSI requirement
IsAuthorizationRequired	Must validate identity of originator against a list of authorized originators

1363

1364 IsAuthorizationRequired is specified on the requesting and
 1365 responding activity accordingly.

1366

1367 6.5.5 Document security

1368 The following security characteristics of each Business Document
 1369 being transported, even if many are collected in the same
 1370 message, can be specified individually, or collectively within a
 1371 Document Envelope:

Parameter	Delivery Channel requirement
<i>isConfidential.</i>	The information entity is encrypted so that unauthorized parties cannot view the information
<i>isTamperProof.</i>	The information entity has an encrypted message digest that can be used to check if the message has been tampered with. This requires a digital signature (sender's digital certificate and encrypted message digest) associated with the document entity.
<i>isAuthenticated.</i>	There is a digital certificate associated with the document entity. This provides proof of the signer's identity.

1372

1373 The value of *isConfidential*, *isTamperProof*, *isAuthenticated* at the
 1374 Document Envelope always applies to the primary Business
 1375 Document. It also applies to each of the attachments unless
 1376 specifically overridden at the Attachment level.

1377 When set to YES (or TRUE) these parameters assume that the
 1378 corresponding security characteristic is provided in a manner
 1379 providing persistence. Compliance requires that the specified
 1380 character of the document survive its reception at a business
 1381 service interface, and persist as the document is archived or
 1382 forwarded.

1383

1384 6.5.6 Reliability

1385

1386

1387

1388

This parameter at the Business Transaction level states whether guaranteed delivery of the transaction's Business Documents is required.

Parameter	Delivery Channel requirement
IsGuaranteedDeliveryRequired	This means that Business Documents transferred are guaranteed (by some delivery channel or other party other than the trading partners) to be delivered

1389

1390

1391

1392

1393

This is a declaration that trading partners must employ only a delivery channel that provides a third-party delivery guarantee, to send Business Documents in the relevant transaction.

1394 6.5.7 Parameters required for CPP/CPA

1395

1396

1397

1398

1399

The ebXML *Business Process Specification Schema* provides parameters that can be used to specify certain levels of security and reliability. The ebXML *Business Process Specification Schema* provides these parameters in general business terms.

1400

1401

1402

1403

These parameters are generic requirements for the business process, but for ebXML implementations, these parameters are specifically used to instruct the CPP and CPA to require BSI and/or delivery channel capabilities to achieve the specified service levels.

1404

The CPP and CPA translate these into parameters of two kinds.

1405

1406

1407

1408

One kind of parameter determines the selection of certain security and reliability parameters applicable to the transport method and techniques used by the delivery channel. Document security, and Reliability above, are determinators of delivery channel selection.

1409

1410

1411

The other kind of parameter determines the selection of certain service levels or capabilities of the BSI itself, in order for it to support the run time Business Transaction semantics as listed below.

1412 **6.6 Run time Business Transaction semantics**

1413

1414

1415

1416

The ebXML concept of a business transaction and the semantics behind it are central to predictable, enforceable commerce. It is expected that any Business Service Interface (BSI) will be capable of managing a transaction according to these semantics.

1417 Therefore, the Business Service Interface (BSI), or any software that implements
1418 one role in an ebXML collaboration needs at minimum to be able to support the
1419 following transaction semantics:

- 1420 1. Detection of the opening of a transaction
- 1421 2. Detection of transfer of control
- 1422 3. Detection of successful completion of a transaction
 - 1423 a. Application of business rules expressed as isPositiveResponse
 - 1424 and transition guardCondition and guardExpression for
 - 1425 determination of success
- 1426 4. Detection of failed completion of a transaction
 - 1427 a. Detection of time-outs
 - 1428 b. Detection of exceptions
 - 1429 c. Application of business rules expressed as isPositiveResponse
 - 1430 and transition guardCondition and guardExpression for
 - 1431 determination of failure
- 1432 5. Notification of failure
- 1433 6. Receipt of notification of failure
- 1434 7. Rollback upon failure (note this is the independent responsibility of each
1435 role, it is not a co-coordinated roll-back, there are no 2-phase commits in
1436 ebXML)

1437 ebXML does not specify how these transaction semantics are implemented but it
1438 is assumed that any Business Service Interface (BSI) will be able to support
1439 these basic transaction semantics at runtime. If either party cannot provide full
1440 support, then the requirements may be relaxed as overrides in the CPP/CPA.

1441 The following sections discuss the two causes of failure: Time-outs and
1442 Exceptions. When either one happens, it is the responsibility of the two roles to
1443 do the necessary roll-back, and to exit the transaction. The responsibilities of the
1444 two roles differ slightly and are described in each of the sections below.
1445 Generally, if a failure happens at the responding role, the responding role will
1446 send an exception signal to the requesting role, and both parties will exit the
1447 current transaction. If a failure happens at the requesting role, the requesting role
1448 will exit the current transaction and in a separate transaction notify the
1449 responding role about the failure. This way the flow of control within a transaction
1450 is always unambiguous and finite.

1451

1452 6.6.1 Timeouts

1453

1454 Since all business transactions must have a distinct time boundary, there
1455 are time-out parameters associated with the response, and each of the
1456 acknowledgement signals. If the time-out occurs before the
1457 corresponding response or signal arrives, the transaction is null and void.

1458

1459

Here are the time-out parameters relative to the three response types:

1460

Response required	Parameter Name	Meaning of timeout
Receipt acknowledgement	timeToAcknowledgeReceipt	The time a responding role has to acknowledge receipt of a business document.
Acceptance Acknowledgement (Non-substantive)	timeToAcknowledgeAcceptance	The time a responding role has to non-substantively acknowledge business acceptance of a business document.
Substantive Response	TimeToPerform	The time a responding role has to substantively acknowledge business acceptance of a business document.

1461

1462

1463

1464

A time-out parameter must be specified whenever a requesting partner expects one or more responses to a business document request. A requesting partner must not remain in an infinite wait state.

1465

1466

1467

1468

The time-out value for each of the time-out parameters is absolute i.e. not relative to each other. All timers start when the initial requesting business document is sent. The timer values must comply with the well-formedness rules for timer values.

1469

1470

A BSI needs to comply with the above parameters to detect the appropriate time outs. To preserve the atomic semantics of the Business

1471 Transaction, the requesting and responding roles take different action
1472 based on time outs.

1473 A responding partner simply terminates if a timeout is thrown. This
1474 prevents responding business transactions from hanging indefinitely.

1475 A requesting partner terminates if a timeout is thrown and then sends a
1476 notification of failure to the responder as part of a separate transaction.

1477 When the time to perform an activity equals the time to acknowledge
1478 receipt or the time to acknowledge business acceptance then the highest
1479 priority time out exception must be used when the originator provides a
1480 reason for revoking their original business document offer. The time to
1481 perform exception is lower priority than both the time to acknowledge
1482 receipt and the time to acknowledge business acceptance.

1483

1484 6.6.2 Exceptions

1485
1486 Under all normal circumstances the response message and/or the time-
1487 outs determine the success or failure of a business transaction. However
1488 the business processing of the transaction can go wrong at either the
1489 responding or the requesting role.

1490 6.6.2.1 ControlException

1491
1492 A *ControlException* signals an error condition in the management of a
1493 business transaction. This business signal is asynchronously returned to
1494 the initiating activity that originated the request. This exception must
1495 terminate the business transaction. These errors deal with the
1496 mechanisms of message exchange such as verification, validation,
1497 authentication and authorization and will occur up to message
1498 acceptance. Typically the rules and constraints applied to the message
1499 will have only dealt with structure, syntax and message element values.

1500 6.6.2.2 Business Protocol Exceptions

1501
1502 A Business Protocol Exception (or *ProcessException*) signals an error
1503 condition in a business activity. This business signal is asynchronously
1504 returned to the initiating role that originated the request. This exception
1505 must terminate the *business transaction*. These errors deal with the
1506 mechanisms that process the *business transaction* and will occur after
1507 message verification and validation. Typically the rules and constraints
1508 applied to the message will deal with the semantics of message elements
1509 and the validity of the request itself. The content is not valid with respect to
1510 a responding role's business rules. This type of exception is usually
1511 generated after an *AcceptanceAcknowledgement* has been returned.

1512 A business protocol exception terminates the business transaction. The
1513 following are business protocol exceptions.

- 1514 • Negative acknowledgement of receipt. The structure/schema of a
1515 message is invalid.
 - 1516 • Negative acknowledgement of acceptance. The business rules
1517 are violated.
 - 1518 • Performance exceptions. The requested business action cannot
1519 be performed.
 - 1520 • Sequence exceptions. The order or type of a business document
1521 or business signal is incorrect.
 - 1522 • Syntax exceptions. There is invalid punctuation, vocabulary or
1523 grammar in the business document or business signal.
 - 1524 • Authorization exceptions. Roles are not authorized to participate in
1525 the business transaction.
 - 1526 • Business process control exceptions. Business documents are not
1527 signed for non-repudiation when required.
- 1528 A Business Transaction is defined in very atomic and deterministic terms.
1529 It always is initiated by the requesting role, and will always conclude at
1530 the requesting role. Upon receipt of the required response and/or signals,
1531 or time-out of same, the requesting role can unambiguously determine
1532 the success or failure of the Business Transaction. To preserve this
1533 semantics, control failures and business failures are treated differently by
1534 the requesting and responding roles as follows:
- 1535 A responding role that encounters a business protocol exception signals
1536 the exception back to the requesting role and then terminates the
1537 business transaction. If any business exceptions (includes negative
1538 receipt and acceptance acknowledgements) are signaled then the
1539 business transaction must terminate.
- 1540 A requesting role that encounters a business protocol exception
1541 terminates the transaction but does NOT send a business exception
1542 signal to the responding role. Rather, the requesting role then sends as a
1543 separate Business Transaction a notification revoking the offending
1544 business document request. This new transaction may be defined as a
1545 continuation of the current Binary Collaboration, or it may start a new
1546 Binary Collaboration specifically defined to handle this notification of
1547 failure.
- 1548 A BSI needs to comply specifically with the following parameters to
1549 produce the associated special exceptions. The requesting and
1550 responding roles take different action as per below.
- 1551 ***IsAuthorizationRequired***
- 1552 If a partner role needs authorization to request a business action
1553 or to respond to a business action then the sending partner role
1554 must sign the business document exchanged and the receiving
1555 partner role must validate this business control and approve the
1556 authorizer. A responding partner must signal an authorization

1557 exception if the sending partner role is not authorized to perform
1558 the business activity. A sending partner must send notification of
1559 failed authorization if a responding partner is not authorized to
1560 perform the responding business activity.

1561 ***IsNonRepudiationRequired***

1562 If non-repudiation of origin and content is required then the
1563 business activity must store the business document in its original
1564 form for the duration mutually agreed to in a trading partner
1565 agreement. A responding partner must signal a business control
1566 exception if the sending partner role has not properly delivered
1567 their business document. A requesting partner must send
1568 notification of failed business control if a responding partner has
1569 not properly delivered their business document.

1570 ***isNonRepudiationOfReceiptRequired.***

1571 Both partners agree to mutually verify receipt of a requesting
1572 business document and that the receipt must be non-repudiatable.
1573 A requesting partner must send notification of failed business
1574 control (possibly revoking a contractual offer) if a responding
1575 partner has not properly delivered their business document. For a
1576 further discussion of nonrepudiation of receipt, see also the
1577 ebXML E-Commerce and Simple Negotiation Patterns

1578

1579 Non-repudiation of receipt provides the data for the following audit
1580 controls.

1581 **Verify responding role identity** (authenticate) – Verify the
1582 identity of the responding role (individual or organization) that
1583 received the requesting business document.

1584 **Verify content integrity** – Verify the integrity of the original
1585 content of the business document request.

1586 ***isPositiveResponse***

1587 An expression whose evaluation results in TRUE or FALSE. If
1588 TRUE this DocumentEnvelope is intended as a positive response
1589 to the request. The value for this parameter supplied for a
1590 DocumentEnvelope is an assertion by the sender of the
1591 DocumentEnvelope regarding its intent for the transaction to
1592 which it relates, but does not bind the recipient, or override the
1593 computation of transactional success or failure using the
1594 transaction's guard expressions.

1595 If a requesting role, upon evaluation of these expressions,
1596 determines a failure, then the requesting role will “roll back” the
1597 Business Transaction and send a notification of failure.

1598

1599

1600 **6.7 Runtime Collaboration Semantics**

1601 The ebXML collaboration semantics contain a number of relationships between
1602 multiparty collaborations and binary collaborations, between recursive layers of
1603 binary collaborations, and choreographies among transactions in binary
1604 collaborations. It is anticipated that over time BSI software will evolve to the point
1605 of monitoring and managing the state of a collaboration, similar to the way a BSI
1606 today is expected to manage the state of a transaction. For the immediate future,
1607 such capabilities are not expected and not required.

1608 **6.8 Where the ebXML Business Process Specification Schema 1609 May Be Implemented**

1610 The ebXML *Business Process Specification Schema* should be used wherever
1611 software is being specified to perform a role in an ebXML business collaboration.
1612 Specifically, the ebXML *Business Process Specification Schema* is intended to
1613 provide the business process and document specification for the formation of
1614 ebXML trading partner Collaboration Protocol Profiles and Agreements.

1615 However, the ebXML *Business Process Specification Schema* may be used to
1616 specify any electronic commerce collaboration. It may also be used for non-
1617 commerce collaborations, for instance in defining transactional collaborations
1618 among non-profit organizations or internally in enterprises.

1619 **7 UML Element Specification**

1620

1621 In the following we will review all the specification elements in the UML version of
1622 the ebXML *Business Process Specification Schema*, grouped as follows:

- 1623 • Business Collaborations
 - 1624 ○ Multiparty
 - 1625 ○ Binary
- 1626 • Business Transactions
- 1627 • Document flow
- 1628 • Choreography

1629

1630 **7.1 Business Collaborations**

1631

1632 **7.1.1 MultiPartyCollaboration**

1633 A Multiparty Collaboration is a synthesis of Binary Collaborations.
1634 A Multiparty Collaboration consists of a number of Business
1635 Partner Roles each playing roles in binary collaborations with
1636 each other.

1637	Tagged Values:	
1638	<i>name.</i>	Defines the name of the MultiPartyCollaboration
1639	Associations:	
1640	<i>partners</i>	A multiparty collaboration has two or more BusinessPartnerRoles
1641		
1642	Wellformedness Rules:	
1643		All multiparty collaborations must be synthesized from binary collaborations
1644		

1645

1646 7.1.2 BusinessPartnerRole

1647 A BusinessPartnerRole is the role played by a business partner in
 1648 a MultiPartyCollaboration. A BusinessPartnerRole performs at
 1649 most one Authorized Role in each of the Binary Collaborations
 1650 that make up the Multiparty Collaboration.

1651	Tagged Values:	
1652	<i>name.</i>	Defines the name of the role played by partner in the overall multiparty business collaboration, e.g. customer or supplier.
1653		
1654		

1655	Associations:	
1656	<i>performers.</i>	The Authorized Roles performed by a partner in the binary business collaboration.
1657		
1658	<i>transitions</i>	The transitions (managed by this BusinessPartnerRole) between activities across binary collaborations
1659		
1660		
1661	<i>collaboration</i>	The Business Partner Role participates in one multi party collaboration
1662		

1663	Wellformedness Rules:	
1664		A partner must not perform both roles in a given business activity.
1665		
1666		

1667 7.1.3 Performs

1668 Performs is an explicit modeling of the relationship between a
 1669 BusinessPartnerRole and the Roles it plays. This specifies the use
 1670 of an Authorized Role within a multiparty collaboration.

1671	Tagged Values:	
1672		
1673	<i>NONE</i>	

1674

Associations:

1675

performedBy An instance of Performs is performed by only one BusinessPartnerRole

1676

1677

authorizedRole The AuthorizedRole that will be performed by the Business PartnerRole

1678

1679

Wellformedness Rules:

1680

For every Performs performing an AuthorizedRole there must be a Performs that performs the opposing AuthorizedRole, otherwise the MultiParty Collaboration is not complete.

1681

1682

1683

1684

1685 **7.1.4 AuthorizedRole**

1686

1687

An Authorized Role is a role that is authorized to send the request or response, e.g. the buyer is authorized to send the request for purchase order, the seller is authorized to send the acceptance of purchase order.

1688

1689

1690

1691

Tagged Values:

1692

name Defines the name of the AuthorizedRole uniquely within the Binary Collaboration

1693

1694

Associations:

1695

performers An AuthorizedRole may be used by one or more performers, i.e. Business Partner Roles in a multiparty collaboration

1696

1697

1698

from An AuthorizedRole may be the initiator in a business activity

1699

1700

to An AuthorizedRole may be the responder in a business activity

1701

1702

collaboration An AuthorizedRole may be in only one BinaryCollaboration

1703

1704

Wellformedness Rules:

1705

An AuthorizedRole may not be both the requestor and the responder in a business transaction

1706

1707

An AuthorizedRole may not be both the initiator and the responder in a binary business collaboration

1708

1709

1710 **7.1.5 BinaryCollaboration**

1711

A Binary Collaboration defines a protocol of interaction between two authorized roles.

1712

1713 A Binary Collaboration is a choreographed set of states among
 1714 collaboration roles. The activities of performing business
 1715 transactions or other collaborations are a kind of state.

1716 A Binary Collaboration choreographs one or more business
 1717 transaction activities between two roles.

1718 A Binary Collaboration is not an atomic transaction and should not
 1719 be used in cases where Business Transaction rollback is required.

1720 Tagged Values:

1721	<i>name</i>	Defines the name of the BinaryCollaboration
1722	<i>timeToPerform</i>	The period of time, starting upon initiation of the
1723		first activity, within which this entire collaboration
1724		must conclude.
1725	<i>preCondition</i>	A description of a state external to this
1726		collaboration that is required before this
1727		collaboration can commence.
1728	<i>postCondition</i>	A description of a state that does not exist
1729		before the execution of this collaboration but will
1730		exist as a result of the execution of this
1731		collaboration.
1732	<i>beginsWhen</i>	A description of an event external to the
1733		collaboration that normally causes this
1734		collaboration to commence.
1735	<i>endsWhen</i>	A description of an event external to this
1736		collaboration that normally causes this
1737		collaboration to conclude.
1738	<i>pattern</i>	The optional reference to a pattern that this
1739		binary collaboration is based on

1740 Associations:

1741	<i>role</i>	A binary collaboration consists of two authorized
1742		roles
1743	<i>states</i>	A binary collaboration consists of one or more
1744		states, some of which are 'static', and some of
1745		which are action states
1746	<i>usedBy</i>	A binary collaboration may be used within
1747		another binary collaboration via a collaboration
1748		activity
1749	<i>transitions</i>	The transitions between activities in this binary
1750		collaboration

1751 Wellformedness Rules:

1752 NONE

1753

1754 **7.1.6 BusinessActivity**1755
1756
1757
1758
1759

A business activity is an action state within a binary collaboration. It is the super type for BusinessTransactionActivity and CollaborationActivity, specifying the activity of performing a transaction or another binary collaboration respectively.

1760
1761**Supertype of:**

BusinessActionActivity, CollaborationActivity

1762
1763**Subtype of:**

BusinessState

1764
1765
1766**Tagged Values:**

name Defines the name of the activity uniquely within the binary collaboration

1767
1768
1769**Associations:**

from The initiating role
to The responding role

1770
1771
1772**Wellformedness Rules:**

NONE

1773 **7.1.7 BusinessTransactionActivity**1774
1775
1776
1777
1778
1779

A business transaction activity defines the use of a business transaction within a binary collaboration.

A business transaction activity is a business activity that executes a specified business transaction. More than one instance of the same business transaction activity can be open at one time if the *isConcurrent* property is *true*.

1780
1781**Subtype of:**

BusinessActivity

1782
1783
1784
1785
1786
1787
1788
1789**Tagged Values:**

timeToPerform The period of time, starting upon the sending of the request, within which both partners agree to conclude the business transaction executed by this Business Transaction Activity.

isConcurrent. If the BusinessTransactionActivity is concurrent then more than one instance of the associated BusinessTransaction can be performed as part

1790 of the execution of this
 1791 BusinessTransactionActivity
 1792 *isLegallyBinding* Defines whether the Business Transaction
 1793 performed by this activity is intended by the
 1794 trading parties to be binding. Default value is
 1795 True.

1796 **Associations:**

1797 *uses.* The business transaction activity performs
 1798 (uses) exactly one business transaction.

1799 **Wellformedness Rules:**

1800 NONE

1801

1802 **7.1.8 CollaborationActivity**

1803 A collaboration activity is the activity of performing a binary
 1804 collaboration within another binary collaboration.

1805 **Subtype of:**

1806 BusinessActivity

1807 **Tagged Values:**

1808 *NONE (other than inherited)*

1809 **Associations:**

1810 *uses* A collaboration activity uses exactly one binary
 1811 collaboration

1812 **Wellformedness Rules:**

1813 A binary collaboration may not re-use itself

1814

1815

1816 **7.2 Business Transactions**

1817

1818 **7.2.1 BusinessTransaction**

1819 A business transaction is a set of business information and
 1820 business signal exchanges amongst two commercial partners that
 1821 must occur in an agreed format, sequence and time period. If any
 1822 of the agreements are violated then the transaction is terminated
 1823 and all business information and business signal exchanges must
 1824 be discarded. Business Transactions can be formal as in the

1825 formation of on-line offer/acceptance commercial contracts and
1826 informal as in the distribution of product announcements.

1827 **Tagged Values:**

1828 *name* Defines the name of the Business Transaction.
1829 *isGuaranteedDeliveryRequired*. Both partners must agree to use
1830 a transport that guarantees delivery
1831 *preCondition* A description of a state external to this
1832 transaction that is required before this
1833 transaction can commence.
1834 *postCondition* A description of a state that does not exist
1835 before the execution of this transaction but will
1836 exist as a result of the execution of this
1837 transaction.
1838 *beginsWhen* A description of an event external to the
1839 transaction that normally causes this transaction
1840 to commence.
1841 *endsWhen* A description of an event external to this
1842 transaction that normally causes this transaction
1843 to conclude.
1844 *pattern* The optional reference to a pattern that this
1845 transaction is based on.

1846 **Associations:**

1847 *activities* A BusinessTransaction can be performed by
1848 many BusinessTransactionActivites
1849 *requester* A BusinessTransaction has exactly one
1850 RequestingBusinessActivity
1851 *responder* A BusinessTransaction has exactly one
1852 RespondingBusinessActivity

1853 **Wellformedness Rules:**

1854 NONE

1855

1856

1857 **7.2.2 Business Action**

1858 A Business Action is an abstract super class. Business Action, is
1859 the holder of attributes that are common to both Requesting
1860 Business Activity and Responding Business Activity.

1861 **Tagged Values:**

1862 *name* Defines the name of the
1863 RequestingBusinessTransaction or

1864 RespondingBusinessTransaction depending on
1865 the subtype

1866 *IsAuthorizationRequired* Receiving party must validate identity
1867 of originator against a list of authorized
1868 originators. This parameter is specified on the
1869 sending side. (See also section on action
1870 security)

1871 *IsNonRepudiationRequired* Receiving party must check that
1872 a requesting document is not garbled
1873 (unreadable, unintelligible) before sending
1874 acknowledgement of receipt. This parameter is
1875 specified on the sending side. (See also section
1876 on core transaction semantics)

1877 *isNonRepudiationOfReceiptRequired.* Requires the receiving
1878 party to return a signed receipt, and the original
1879 sender to save copy of the receipt. This
1880 parameter is specified on the sending side.
1881 (See also section on nonrepudiation)

1882 *timeToAcknowledgeReceipt* The time a receiving role has to
1883 acknowledge receipt of a business document.
1884 This parameter is specified on the sending side.
1885 (See also section on core transaction semantics)

1886 *isIntelligibleCheckRequired* Receiving party must check that
1887 a requesting document is not garbled
1888 (unreadable, unintelligible) before sending
1889 acknowledgement of receipt. This parameter is
1890 specified on the sending side. (See also section
1891 on core transaction semantics)

1892 **Associations:**

1893 *NONE*

1894 **Wellformedness Rules:**

1895 *NONE*

1896

1897

1898 7.2.3 RequestingBusinessActivity

1899 A RequestingBusinessActivity is a Business Action that is
1900 performed by the requesting role within a Business Transaction. It
1901 specifies the Document Envelope which will carry the request.

1902 **Subtype of:**

1903 BusinessAction

1904 **Tagged Values:**

1905 *timeToAcknowledgeAcceptance* The time a responding role has
 1906 to non-substantively acknowledge business
 1907 acceptance of a business document. This
 1908 parameter is specified on the requesting side.
 1909 (See also section on core transaction semantics)

1910 **Associations:**

1911 *transaction* A requesting activity is performed in exactly one
 1912 business transaction

1913 *documentEnvelope* A requesting activity sends exactly one
 1914 Document Envelope

1915

1916 **Wellformedness Rules:**

1917 NONE

1918

1919 7.2.4 RespondingBusinessActivity

1920 A RespondingBusinessActivity is a Business Action that is
 1921 performed by the responding role within a Business Transaction.
 1922 It specifies the Document Envelope which will carry the response.

1923 There may be multiple possible response Document Envelopes
 1924 defined, but only one of them will be sent during an actual
 1925 transaction instance.

1926 **Subtype of:**

1927 BusinessAction

1928 **Tagged Values:**

1929 NONE, except as inherited from Business Action

1930 **Associations:**

1931 *transaction* A responding activity is performed in exactly one
 1932 business transaction

1933 *DocumentEnvelope* A responding activity may specify zero
 1934 or more but sends at most one Document
 1935 Envelope

1936

1937 **Wellformedness Rules:**

1938 NONE

1939

1940 **7.3 Document flow**

1941

1942 **7.3.1 Document Security**

1943 DocumentSecurity is an abstract super class holding the security
1944 related attributes for DocumentEnvelope and Attachment.

1945 **Supertype of:**

1946 DocumentEnvelope and Attachment

1947 **Tagged Values:**

1948 *IsAuthenticated* There is a digital certificate associated with the
1949 document entity. This provides proof of the
1950 signer's identity. (See also section on Document
1951 Security)

1952 *IsConfidential* The information entity is encrypted so that
1953 unauthorized parties cannot view the
1954 information. (See also section on Document
1955 Security)

1956 *isTamperProof* The information entity has an encrypted
1957 message digest that can be used to check if the
1958 message has been tampered with. This requires
1959 a digital signature (sender's digital certificate
1960 and encrypted message digest) associated with
1961 the document entity. (See also section on
1962 Document Security)

1963 **Associations:**

1964 NONE

1965 **Wellformedness Rules:**

1966 NONE

1967 **7.3.2 Document Envelope**

1968 A Document Envelope is what conveys business information
1969 between the two roles in a business transaction. One Document
1970 Envelope conveys the request from the requesting role to the
1971 responding role, and another Document Envelope conveys the
1972 response (if any) from the responding role back to the requesting
1973 role.

1974 **Subtype of:**

1975 DocumentSecurity

1976

1977 **Tagged Values:**

1978	<i>isPositiveResponse</i>	An expression whose evaluation results in TRUE or FALSE. If TRUE this DocumentEnvelope is intended as a positive response to the request. This parameter is only relevant on the response envelope. Its value does not bind the recipient, or override the computation of transactional success or failure using the transaction's guard expressions.
1979		
1980		
1981		
1982		
1983		
1984		
1985		
1986	Associations:	
1987	<i>requesting</i>	This is a reference to the requesting activity associated with this DocumentEnvelope. This requesting activity may be the sender, or the receiver depending on whether the DocumentEnvelope represents a request or a response.
1988		
1989		
1990		
1991		
1992		
1993	<i>responding</i>	This is a reference to the requesting activity associated with this DocumentEnvelope. This responding activity may be the sender, or the receiver depending on whether the DocumentEnvelope represents a request or a response.
1994		
1995		
1996		
1997		
1998		
1999	<i>BusinessDocument</i>	This identifies the primary Business Document in the envelope. A Document Envelope contains exactly one primary Business Document.
2000		
2001		
2002		
2003	<i>attachment</i>	A Document Envelope contains an optional set of attachments related to the primary document
2004		
2005	Wellformedness Rules:	
2006		A Document Envelope is associated with exactly one requesting and one responding activity.
2007		
2008		<i>isPositiveResponse</i> is not a relevant parameter on a DocumentEnvelope sent by a requesting activity
2009		
2010		
2011		
2012		
2013		
2014		
2015		
2016	7.3.3 BusinessDocument	
2017		BusinessDocument is a generic name of a document. The location of the definition of the document can be found in the associated DocumentSpecification.
2018		
2019		
2020	Tagged Values:	

2021 *name* Defines the generic name of the Business
 2022 Document as it is known within this Business
 2023 Process Specification

2024 **Associations:**

2025 *documentSpecification* A Business Document is in at most one
 2026 DocumentSpecification

2027 *documentEnvelope* A Business Document can be in multiple
 2028 Document Envelopes

2029 *attachment* A Business Document can serve to specify the
 2030 type of many attachments

2031 **Wellformedness Rules:**

2032 NONE

2033 **7.3.4 DocumentSpecification**

2034 A DocumentSpecification is a collection of Document Definitions.
 2035 The DocumentSpecification is usually external to the process
 2036 specification, and is referenced with a URI. An additional
 2037 reference is to where the logical model is for the Documents in the
 2038 DocumentSpecification. Typically this would be an ebXML core
 2039 component context model.

2040 **Tagged Values:**

2041 *name* Defines the generic name of the
 2042 DocumentSpecification as it is known within this
 2043 Business Process Specification

2044 *location* Reference to an external source of the
 2045 DocumentSpecification definition

2046 *logicalModel* Reference is to where the logical model is

2047 *version* The version of the DocumentSpecification

2048

2049 **Associations:**

2050 *businessDocument* A DocumentSpecification defines many
 2051 document types

2052 **Wellformedness Rules:**

2053 NONE

2054

2055 **7.3.5 Attachment**

2056 Attachment is an optional attachment to a BusinessDocument in a
 2057 Document Envelope

2058	Subtype of:	
2059		DocumentSecurity
2060	Tagged Values:	
2061	<i>name</i>	Defines the name of the attachment
2062	<i>mimeType</i>	Defines the valid MIME (Multipurpose Internet Mail Extensions) type of this Attachment
2063		
2064	<i>specification</i>	A reference to an external source of description of this attachment.
2065		
2066	<i>version</i>	The version of the Attachment
2067	Associations:	
2068	<i>documentEnvelope</i>	An Attachment is in exactly one Document Envelope
2069		
2070	<i>businessDocument</i>	An Attachment can be defined by a BusinessDocument. If it is not of a defined Business Document, the mime type and spec will be the only indication of its type.
2071		
2072		
2073		
2074	Wellformedness Rules:	
2075		NONE
2076		
2077		

2078 **7.4 Choreography within Collaborations.**

2079
2080

2081 **7.4.1 BusinessState**

2082 A business state is any state that a binary collaboration can be in.
2083 Start and CompletionState are a snapshot right before or right
2084 after an activity, BusinessActivity is an action states that denote
2085 the state of being in an activity. Fork and Join reflect the activity of
2086 forking to multiple activities or joining back from them.

2087 **Supertype of:**
2088 Start, CompletionState, Fork, Join, BusinessActivity

2089 **Tagged Values:**
2090 *none*

2091 **Associations:**
2092 *collaboration* A business state belongs to only one binary
2093 collaboration

2094	<i>entering</i>	A transition that reflects entry into this state
2095	<i>exiting</i>	A transition that reflects exiting from this state
2096	Wellformedness Rules:	
2097	NONE	
2098		
2099	7.4.2 Transition	
2100	A transition is a transition between two business states in a binary	
2101	collaboration.	
2102	Choreography is expressed as transitions between business	
2103	states	
2104	Tagged Values:	
2105	<i>onInitiation</i>	This specifies this is a nested
2106		BusinessTransactionActivity and that upon
2107		receipt of the request in the associated
2108		transaction a second activity is performed before
2109		returning to the transaction to send the response
2110		back to the original requestor.
2111	<i>guardCondition</i>	A reference to the status of the previous
2112		transaction. A fixed value of Success,
2113		BusinessFailure, TechnicalFailure, or AnyFailure
2114	<i>guardExpression</i>	An expression whose evaluation results in
2115		TRUE or FALSE to determine if this transition
2116		should happen or not. The expression can refer
2117		to the name or content of the the most recent
2118		DocumentEnvelope or content of documents
2119		within it.
2120	Associations:	
2121	<i>in</i>	The business state this transition is entering
2122	<i>out</i>	The business state this transition is exiting
2123	Wellformedness Rules:	
2124	A transition cannot enter and exit the same state	
2125		
2126	7.4.3 Start	
2127	The starting state for an Binary Collaboration. A Binary	
2128	Collaboration should have at least one starting activity. If none	
2129	defined, then all activities are considered allowable entry points.	
2130	Subtype of:	
2131	BusinessState	

2132	Tagged Values:
2133	<i>NONE</i>
2134	Associations:
2135	<i>NONE</i>
2136	Wellformedness Rules:
2137	<i>NONE</i>
2138	7.4.4 CompletionState
2139	The ending state of an binary collaboration, sub classed by
2140	success and failure
2141	Supertype of:
2142	Success, Failure
2143	Subtype of:
2144	BusinessState
2145	CompletionState
2146	Tagged Values:
2147	<i>NONE</i>
2148	Associations:
2149	<i>NONE</i>
2150	Wellformedness Rules:
2151	<i>NONE</i>
2152	7.4.5 Success
2153	Defines the successful conclusion of a binary collaboration as a
2154	transition from an activity.
2155	Subtype of:
2156	CompletionState
2157	Tagged Values:
2158	<i>NONE</i> , except as inherited
2159	Associations:
2160	<i>NONE</i>
2161	Wellformedness Rules:

2162		Every activity Binary Collaboration should have at least one
2163		success
2164		
2165	7.4.6 Failure	
2166		A subtype of CompletionState which defines the unsuccessful
2167		conclusion of a binary collaboration as a transition from an activity.
2168		Subtype of:
2169		CompletionState
2170		Tagged Values:
2171		<i>NONE</i> , except as inherited
2172		Associations:
2173		<i>None</i>
2174		Wellformedness Rules:
2175		Every Binary Collaboration should have at least one failure
2176	7.4.7 Fork	
2177		A Fork is a state with one inbound transition and multiple
2178		outbound transitions. All activities pointed to by the outbound
2179		transitions are assumed to happen in parallel.
2180		Subtype of:
2181		BusinessState
2182		Tagged Values:
2183		<i>Name</i> Defines the name of the Fork state
2184		Associations:
2185		<i>None</i>
2186		Wellformedness Rules:
2187		<i>None</i>
2188		
2189	7.4.8 Join	
2190		A business state where an activity is waiting for the completion of
2191		one or more other activities. Defines the point where previously
2192		forked activities join up again.
2193		Subtype of:
2194		BusinessState

2195	Tagged Values:	
2196	<i>Name</i>	Defines the name of the Join state
2197	<i>waitForAll</i>	Boolean value indicating if this Join state should
2198		wait for all incoming transitions to complete. If
2199		TRUE, wait for all, if False proceed on first
2200		incoming transition.

2201	Associations:	
2202	<i>None</i>	

2203	Wellformedness Rules:	
2204	<i>None</i>	

2205
2206
2207

2208 **7.5 Definition and Scope**

2209 The ebXML *Business Process Specification Schema* should be used wherever
2210 software is being specified to perform a role in an ebXML binary collaboration.
2211 Specifically, the ebXML *Business Process Specification Schema* is intended to
2212 provide the business process and document specification for the formation of a
2213 trading partner Collaboration Protocol Profile and Agreement. A set of
2214 specification rules have been established to properly constrain the expression of
2215 a business process and information model in a way that can be directly
2216 incorporated into a trading partner Collaboration Protocol Profile and Agreement.

2217 **7.6 Collaboration and transaction well-formedness rules**

2218 The following rules should be used in addition to standard parsing to properly
2219 constrain the values of the attributes of the elements in an ebXML Business
2220 Process Specification.

2221 *Business Transaction*

- 2222 [0] If non-repudiation is required then the input or returned business
2223 document must be a tamper-proofed entity.
- 2224 [1] If authorization is required then the input business document and
2225 business signal must be an authenticated or a tamper proofed secure
2226 entity.
- 2227 [2] The time to acknowledge receipt must be less than the time to
2228 acknowledge acceptance if both properties have values.
- 2229 $timeToAcknowledgeReceipt < timeToAcknowledgeAcceptance$
2230

- 2231 [3] If the time to acknowledge acceptance is null then the time to perform
2232 an activity must either be equal to or greater than the time to
2233 acknowledge receipt.
- 2234 [4] The time to perform a transaction cannot be null if either the time to
2235 acknowledge receipt or the time to acknowledge acceptance is not
2236 null.
- 2237 [5] If non-repudiation of receipt is required then the time to acknowledge
2238 receipt cannot be null.
- 2239 [6] The time to acknowledge receipt, time to acknowledge acceptance
2240 and time to perform cannot all be zero.
- 2241 [7] If non-repudiation is required at the requesting business activity, then
2242 there must be a responding business document.
- 2243 [8] The time to acknowledge receipt, time to acknowledge acceptance
2244 and time to perform properties must be specified for both the
2245 requesting and responding business activities and they must be
2246 equal.
- 2247 *RequestingBusinessActivity*
- 2248 [9] There must be one input transition whose source state vertex is an
2249 initial pseudo state.
- 2250 [10] There must be one output transition whose target state vertex is a
2251 final state specifying the state of the machine when the activity is
2252 successfully performed.
- 2253 [11] There must be one output transition whose target state vertex is a
2254 final state specifying the state of the machine when the activity is NOT
2255 successfully performed due to a process control exception.
- 2256 [12] There must be one output transition whose target state vertex is a
2257 final state specifying the state of the machine when the activity is NOT
2258 successfully performed due to a business process exception.
- 2259 [13] There must be one output document flow from a requesting
2260 business activity that in turn is the input to a responding business
2261 activity.
- 2262 [14] There must be zero or one output document flow from a
2263 responding business activity that in turn is the input to the requesting
2264 business activity.
- 2265 *RespondingBusinessActivity*
- 2266 [15] There must be one input transition from a document flow that in
2267 turn has one input transition from a requesting business activity.
- 2268 [16] There must be zero or one output transition to an document flow
2269 that in turn has an output transition to a requesting business activity.

2270

Business Collaboration

2271

[17] A Business Partner Role cannot provide both the initiating and responding roles of the same business transaction activity.

2272

2273 8 ebXML Business Process Specification Schema – 2274 (DTD)

2275 In this section we describe the DTD and XML Schema version of the
2276 Specification Schema. There are minimal differences between the DTD and the
2277 XML Schema, therefore the elements will only be described once, noting
2278 differences when needed. This discussion includes

- 2279 • An example XML Business Process Specification listed in Appendix A.
- 2280 • A listing of the DTD in Appendix B and the XML Schema in Appendix C
- 2281 • A table listing all the elements with definitions and parent/child
2282 relationships
- 2283 • A table listing all the attributes with definitions and parent element
2284 relationships
- 2285 • A table listing all the elements, each with a cross reference to the
2286 corresponding class in the UML version of the specification schema
- 2287 • Rules about namespaces and element references

2288 8.1 Documentation for the DTD

2289 This section will document the DTD. The DTD has been derived from the UML
2290 model. The correlation between the UML classes and DTD elements will be
2291 shown separately later in this document.
2292

2293 Overall Structure excluding attribute definitions:

2294 [ProcessSpecification](#) (Documentation*, (Include* | DocumentSpecification* |
2295 ProcessSpecification* | Package | BinaryCollaboration |
2296 BusinessTransaction | MultiPartyCollaboration)*)
2297 [Documentation](#)()
2298 [Include](#)(Documentation*)
2299 [DocumentSpecification](#)(Documentation*, BusinessDocument*)
2300 [BusinessDocument](#)(Documentation*)
2301 [Package](#)(Documentation*, (Package | BinaryCollaboration |
2302 BusinessTransaction | MultiPartyCollaboration)*)
2303 [BinaryCollaboration](#)(Documentation*, AuthorizedRole, AuthorizedRole,
2304 (Documentation* | Start | Transition | Success | Failure |
2305 BusinessTransactionActivity | CollaborationActivity | Fork | Join)*)
2306 [AuthorizedRole](#)(Documentation*)
2307 [Start](#)(Documentation*)
2308 [Transition](#)(Documentation*)
2309 [Success](#)(Documentation*)
2310 [Failure](#)(Documentation*)
2311 [Fork](#)(Documentation*)
2312 [Join](#)(Documentation*)
2313 [BusinessTransactionActivity](#)(Documentation*)
2314 [CollaborationActivity](#)(Documentation*)
2315 [BusinessTransaction](#)(Documentation*, RequestingBusinessActivity,

2316 RespondingBusinessActivity)
 2317 [RequestingBusinessActivity](#)(Documentation*, DocumentEnvelope)
 2318 [RespondingBusinessActivity](#)(Documentation*, DocumentEnvelope*)
 2319 [MultiPartyCollaboration](#)(Documentation*, BusinessPartnerRole*)
 2320 [BusinessPartnerRole](#)(Documentation*, Performs*, Transition*)
 2321 [Performs](#)(Documentation*)
 2322 [Transition](#)(Documentation*)
 2323
 2324

a. Attachment

2325 **XML Element Name:** Attachment

2326 **DTD Declaration:**

```
2327 <!ELEMENT Attachment (Documentation*)>
2328 <!ATTLIST Attachment
2329     name                CDATA #REQUIRED
2330     nameID              ID #IMPLIED
2331     BusinessDocument    CDATA #IMPLIED
2332     BusinessDocumentIDRef IDREF #IMPLIED
2333     mimeType            CDATA #IMPLIED
2334     specification       CDATA #IMPLIED
2335     version             CDATA #IMPLIED
2336     isConfidential      (true | false) "false"
2337     isTamperProof      (true | false) "false"
2338     isAuthenticated     (true | false) "false">
```

2339

Definition:

2340 An optional attachment to a BusinessDocument in a DocumentEnvelope.

2341

2342 **Parent Elements:**

2343

- DocumentEnvelope

2344

Attributes:

Attribute Name	Definition	Default Value
name	Defines the name of the attachment.	Required input
nameID	XML ID version of name	Optional input
businessDocument	An Attachment can be defined by a BusinessDocument. If it is not of a defined Business Document, the mime type and spec will be the only indication of its type.	Required input
businessDocumentIDRef	The XML IDREF version of businessDocument	Optional input
isAuthenticated	There is a digital certificate associated with the document entity. This provides proof of the signer's identity. (See also section on Document Security)	false {true, false}
isConfidential	The information entity is encrypted so that unauthorized parties cannot view the information (See also section on Document Security)	false {true, false}
isTamperProof	The information entity has an encrypted message digest that can be used to check if the message has been tampered with. This requires a digital signature (sender's digital certificate and encrypted message digest) associated with the document entity. (See also section on Document Security)	false Valid values {true, false}
mimeType	Defines the valid MIME (Multipurpose Internet Mail Extensions) type of this Attachment	Optional input. Example: 'application/pdf'
specification	A reference to an external source of description of this attachment.	Optional Input
version	The version of the Attachment	Optional Input

2345

2346

b. Authorized Role

2347

XML Element Name: AuthorizedRole

2348

DTD Declaration:

2349

`<!ELEMENT AuthorizedRole (Documentation*)>`

2350

`<!ATTLIST AuthorizedRole`

2351

`name CDATA #REQUIRED`

2352

`nameID ID #IMPLIED>`

2353

2354

Definition:

2355

An Authorized Role is a role that is authorized to send the request or response, e.g. the buyer is authorized to send the request for purchase order, the seller is authorized to send the acceptance of purchase order.

2356

2357

2358

2359

Parents:

2360

BinaryCollaboration

2361

Attributes:

Attribute Name	Definition	Default Value
Name	Defines the name of the Authorized Role	Required input.
nameID	XML ID version of name	Optional

2362

2363

c. Binary Collaboration

2364

XML Element Name: BinaryCollaboration

2365

DTD Declaration:

2366

`<!ELEMENT BinaryCollaboration (Documentation*,`

2367

`AuthorizedRole, AuthorizedRole,(Documentation* | Start |`

2368

`Transition | Success | Failure |`

2369

`BusinessTransactionActivity | CollaborationActivity | Fork`

2370

`| Join)*>`

2371

`<!ATTLIST BinaryCollaboration`

2372

`name CDATA #REQUIRED`

2373

`nameID ID #IMPLIED`

2374

`pattern CDATA #IMPLIED`

2375

`beginsWhen CDATA #IMPLIED`

2376

`endsWhen CDATA #IMPLIED`

2377

`precondition CDATA #IMPLIED`

2378

`postCondition CDATA #IMPLIED`

2379

`timeToPerform CDATA #IMPLIED`

2380

`>`

2381

Definition:

2382

A Binary Collaboration defines a protocol of interaction between two authorized roles.

2383

2384

A Binary Collaboration is a choreographed set of states among collaboration roles. The activities of performing business transactions or other collaborations are a kind of state.

2385

2386

2387 A Binary Collaboration choreographs one or more business transaction activities
2388 between two roles.

2389 A Binary Collaboration is not an atomic transaction and should not be used in
2390 cases where Business Transaction rollback is required.

2391

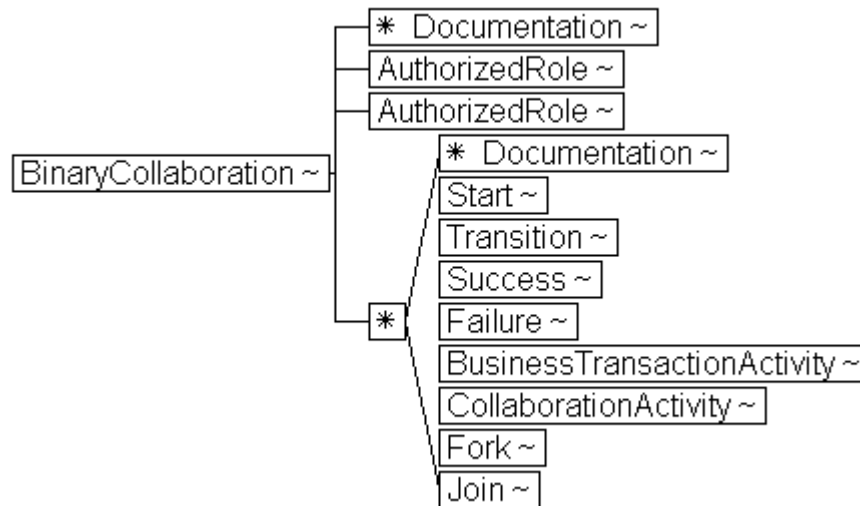
2392 **Parents:**

- Package

2394

2395

Hierarchical Model:



2396

2397

2398

Attributes:

Attribute Name	Definition	Default Value
name	Defines the name of a model element. This name must be unique within the context of the model element and will be used to reference the element from other points in the model.	Required Input.
nameID	The XML ID version of name	Optional
beginsWhen	A description of an event external to the collaboration that normally causes this collaboration to commence.	Optional Input.
endsWhen	A description of an event external to this collaboration that normally causes this collaboration to conclude.	Optional Input.
pattern	The optional reference to a pattern that this binary collaboration is based on.. In the XML Schema version the data type is xsd:anyURI	

preCondition	A description of a state external to this collaboration that is required before this collaboration can commence.	Optional Input.
postCondition	A description of a state that does not exist before the execution of this collaboration but will exist as a result of the execution of this collaboration.	Optional Input..
timeToPerform	The period of time, starting upon initiation of the first activity, within which this entire collaboration must conclude.	Optional Input.

2399

2400

2401

2402

c: Business Partner Role

2403

2404

Element Name: BusinessPartnerRole

2405

DTD Declaration:

2406

```
<!ELEMENT BusinessPartnerRole (Documentation*, Performs*,
                                Transition*)>
```

2407

2408

```
<!ATTLIST BusinessPartnerRole
            name CDATA #REQUIRED
            nameID ID #IMPLIED>
```

2409

2410

2411

2412

Definition:

2413

A BusinessPartnerRole is the role played by a business partner in a MultiPartyCollaboration. A BusinessPartnerRole performs at most one Authorized Role in each of the Binary Collaborations that make up the Multiparty Collaboration.

2414

2415

2416

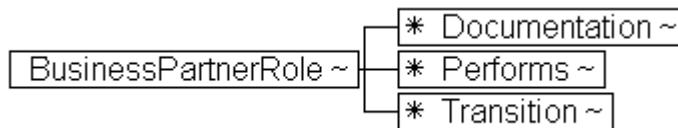
2417

Parents:

2418

- MultiPartyCollaboration

2419

Hierarchical Model:

2420

2421

2422

Attributes:

Attribute Name	Definition	Default Value
Name	Defines the name of the role played by partner in the overall multiparty business collaboration, e.g. customer or supplier.	Required Input.

nameID	The XML ID version of name	Optional
---------------	----------------------------	----------

2423

2424

d. Business Transaction

2425

Element Name: BusinessTransaction

2426

Content Model:

2427

```
<!ELEMENT BusinessTransaction (Documentation*,
RequestingBusinessActivity, RespondingBusinessActivity)>
```

2428

2429

```
<!ATTLIST BusinessTransaction
```

2430

```
    name                CDATA #REQUIRED
```

2431

```
    nameID              ID #IMPLIED
```

2432

```
    pattern              CDATA #IMPLIED
```

2433

```
    beginsWhen          CDATA #IMPLIED
```

2434

```
    endsWhen            CDATA #IMPLIED
```

2435

```
    isGuaranteedDeliveryRequired (true | false) false
```

2436

```
    precondition        CDATA #IMPLIED
```

2437

```
    postCondition        CDATA #IMPLIED>
```

2438

2439

Definition:

2440

A business transaction is a set of business information and business signal exchanges amongst two commercial partners that must occur in an agreed format, sequence and time period. If any of the agreements are violated then the transaction is terminated and all business information and business signal exchanges must be discarded. Business Transactions can be formal as in the formation of on-line offer/acceptance commercial contracts and informal as in the distribution of product announcements.

2441

2442

2443

2444

2445

2446

2447

2448

2449

2450

Parents:

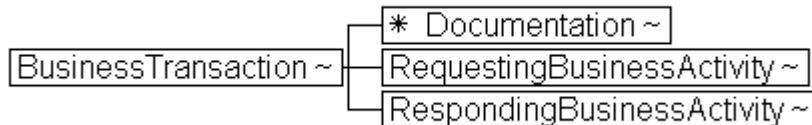
2451

- Package

2452

2453

Hierarchical Model:



2454

2455

2456

Attributes:

2457

Attribute Name	Definition	Default Value
name	Defines the name of the Business Transaction.	Required Input.

nameID	The XML ID version of name	Optional
pattern	The optional reference to a pattern that this transaction is based on. In the XML Schema version the data type is xsd:anyURI	Optional
beginsWhen	A description of an event external to the transaction that normally causes this transaction to commence.	Optional Input..
endsWhen	A description of an event external to this transaction that normally causes this transaction to conclude.	Optional Input.
isGuaranteedDeliveryRequired	Both partners must agree to use a transport that guarantees delivery	false Valid Values: {true, false}
preCondition	A description of a state external to this transaction that is required before this transaction can commence.	Optional Input.
postCondition	A description of a state that does not exist before the execution of this transaction but will exist as a result of the execution of this transaction.	Optional Input.

2458

2459

e. Business Transaction Activity

2460

Element Name: BusinessTransactionActivity

2461

Content Model:

2462

`<!ELEMENT BusinessTransactionActivity (Documentation*)>`

2463

`<!ATTLIST BusinessTransactionActivity`

2464

`name CDATA #REQUIRED`

2465

`nameID ID #IMPLIED`

2466

`businessTransaction CDATA #REQUIRED`

2467

`businessTransactionIDRef IDREF #IMPLIED`

2468

`fromAuthorizedRole CDATA #REQUIRED`

2469

`fromAuthorizedRoleIDRef IDREF #IMPLIED`

2470

`toAuthorizedRole CDATA #REQUIRED`

2471

`toAuthorizedRoleIDRef IDREF #IMPLIED`

2472

`isConcurrent (true | false) "false"`

2473

`isLegallyBinding (true | false) "true"`

2474

`timeToPerform CDATA #IMPLIED>`

2475

Definition:

2476 A business transaction activity defines the use of a business transaction within a
2477 binary collaboration.

2478 A business transaction activity is a business activity that executes a specified
2479 business transaction. More than one instance of the same business transaction
2480 activity can be open at one time if the isConcurrent property is true.

2481 **Parents:**

- 2482 • BinaryCollaboration

2483 **Attributes:**

Attribute Name	Definition	Default Value
name	Defines the name of the activity uniquely within the binary collaboration	Required Input.
nameID	The XML ID version of name	Optional Input
businessTransaction	A reference, by name to the Business Transaction performed by this Business Transaction Activity	Required Input.
businessTransactionIDRef	The XML IDREF version of businessTransaction	Optional Input.
fromAuthorizedRole	The name of the initiating role in Business Transaction Activity. This must match one of the AuthorizedRoles in the binary collaboration and will become the requestor in the BusinessTransaction performed by this activity	Required Input.
fromAuthorizedRoleIDRef	The XML IDREF version of fromAuthorizedRole	Optional Input.
toAuthorizedRole	The name of the responding role in Business Transaction Activity. This must match one of the AuthorizedRoles in the binary collaboration and will become the responder in the BusinessTransaction performed by this activity	Required Input.
toAuthorizedRoleIDRef	The XML IDREF version of toAuthorizedRole	Optional Input.
timeToPerform	The period of time, starting upon the sending of the request, within which both partners agree to conclude the business transaction executed by this Business Transaction Activity.	Optional Input.

isLegallyBinding	Defines whether the Business Transaction performed by this activity is intended by the trading parties to be binding. Default value is True.	true Valid Values: {true, false}
isConcurrent	If the BusinessTransactionActivity is concurrent then more than one instance of the associated BusinessTransaction can be open the same time as part of the execution of this BusinessTransactionActivity	false Valid Values: {true, false}

2484

2485

f. Collaboration Activity

2486

Element Name: CollaborationActivity

2487

DTD Declaration:

2488

```
<!ELEMENT CollaborationActivity (Documentation*)>
```

2489

```
<!ATTLIST CollaborationActivity
```

2490

```
  name CDATA #REQUIRED
```

2491

```
  nameID ID #IMPLIED
```

2492

```
  fromAuthorizedRole CDATA #REQUIRED
```

2493

```
  fromAuthorizedRoleIDRef CDATA #IMPLIED
```

2494

```
  toAuthorizedRole CDATA #REQUIRED
```

2495

```
  toAuthorizedRoleIDRef CDATA #IMPLIED
```

2496

```
  binaryCollaboration CDATA #REQUIRED>
```

2497

```
  binaryCollaborationIDRef CDATA #IMPLIED>
```

2498

Definition:

2499

A collaboration activity is the activity of performing a binary collaboration within another binary collaboration.

2500

2501

2502

Parents:

2503

- BinaryCollaboration

2504

Attributes:

Attribute Name	Definition	Default Value
name	Defines the name of the activity uniquely within the binary collaboration	Required Input.
fromAuthorizedRole	The name of the initiating role in the Collaboration Activity. This must match one of the AuthorizedRoles in the parent binary collaboration and will become the initiator in the BinaryCollaboration performed by this activity	Required Input

fromAuthorizedRoleIDRef	The XML IDREF version of fromAuthorizedRole	OptionalInput.
toAuthorizedRole	The name of the responding role in the Collaboration Activity. This must match one of the AuthorizedRoles in the parent binary collaboration and will become the responder in the BinaryCollaboration performed by this activity	Required Input.
toAuthorizedRoleIDRef	The XML IDREF version of toAuthorizedRole	Optional Input.
binaryCollaboration	A reference, by name, to the Binary Collaboration performed by this Collaboration Activity	Required Input.
binaryCollaborationIDRef	The XML IDREF version of binaryCollaboration	Optional Input.

2505

2506

g. Documentation

2507

Element Name: Documentation

2508

DTD Declaration:

2509

`<!ELEMENT Documentation (#PCDATA)>`

2510

`<!ATTLIST Documentation`

2511

`uri CDATA #IMPLIED>`

2512

2513

Definition:

2514

Defines user documentation for any element. Must be the first element of its container. Documentation can be either inline PCDATA and/or a URI to where more complete documentation is to be found

2515

2516

2517

Parents:

2518

- AuthorizedRole

2519

- BinaryCollaboration

2520

- BusinessPartnerRole

2521

- BusinessTransaction

2522

- BusinessTransactionActivity

2523

- CollaborationActivity

2524

- DocumentEnvelope

2525

- BusinessDocument

2526

- ProcessSpecification

2527

- MultiPartyCollaboration

2528

- Package

2529

- Performs

2530

- RequestingBusinessActivity

- 2531 • RespondingBusinessActivity
 2532 • DocumentSpecification
 2533 • Transition
 2534 **Attributes:**
 2535

Attribute Name	Definition	Default Value
uri	Defines the URI (Uniform Resource Identifier) where external documentation is located. In the XML Schema version the data type is xsd:anyURI	No Default Value. Valid URI is required.

2536

2537

h. DocumentEnvelope

2538

Element Name: DocumentEnvelope

2539

Content Model:

2540

```
<!ELEMENT DocumentEnvelope (Documentation*,
2541 BusinessDocument,
2542 Attachment*)>
```

2543

```
<!ATTLIST DocumentEnvelope
2544 isPositiveResponse CDATA #IMPLIED
2545 isAuthenticated (true | false) "false"
2546 isConfidential (true | false) "false"
2547 isTamperProof (true | false) "false">
```

2548

2549

Definition:

2550

A DocumentEnvelope is what conveys business information between the two roles in a business transaction. One DocumentEnvelope conveys the request from the requesting role to the responding role, and another DocumentEnvelope conveys the response (if any) from the responding role back to the requesting role.

2551

2552

2553

2554

2555

Parents:

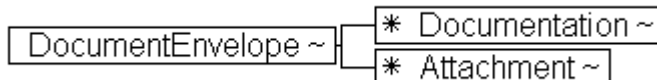
2556

- RequestingBusinessActivity
- RespondingBusinessActivity

2557

2558

2559

Hierarchical Model:

2560

2561

2562

Attributes:

Attribute Name	Definition	Default Value
----------------	------------	---------------

isPositiveResponse	An expression whose evaluation results in TRUE or FALSE. If TRUE this DocumentEnvelope is intended as a positive response to the request. The value for this parameter supplied for a DocumentEnvelope is an assertion by the sender of the DocumentEnvelope regarding its intent for the transaction to which it relates, but does not bind the recipient, or override the computation of transactional success or failure using the transaction's guard expressions. In some situations this could be an XPath expression that interrogates the BusinessDocument in the envelope.	Optional Input.
isAuthenticated	There is a digital certificate associated with the document entity. This provides proof of the signer's identity. (See also section on Document Security)	false Valid Values: {true, false}
isConfidential	The information entity is encrypted so that unauthorized parties cannot view the information. (See also section on Document Security)	false Valid Values: {true, false}
isTamperProof	The information entity has an encrypted message digest that can be used to check if the message has been tampered with. This requires a digital signature (sender's digital certificate and encrypted message digest) associated with the document entity. (See also section on Document Security)	false Valid Values: {true, false}

2563

2564

2565

2566

i. BusinessDocument

Element Name: BusinessDocument

2567

DTD Declaration:

2568

```
<!ELEMENT BusinessDocument (Documentation*) >
```

2569

```
<!ATTLIST BusinessDocument
```

2570

```
  name CDATA #REQUIRED
```

2571

```
  nameID ID #IMPLIED >
```

2572

Definition:

2573

BusinessDocument is a generic name of a document. The location of the definition of the document can be found in the associated DocumentSpecification.

2574

2575

2576

Parents:

2577

- Attachment

2578

- DocumentSpecification

2579

Attributes:

2580

2581

Attribute Name	Definition	Default Value
name	Defines the generic name of the Business Document as it is known within this Business Process Specification	Required Input
nameID	XML ID version of name	Optional

2582

2583

j. DocumentSpecification

2584

Element Name:

2585

DTD Declaration:

2586

```
<!ELEMENT DocumentSpecification (Documentation*,
BusinessDocument)>
```

2587

```
<!ATTLIST DocumentSpecification
```

2588

```
  name CDATA #REQUIRED
```

2589

```
  nameID ID #IMPLIED
```

2590

```
  location CDATA #IMPLIED
```

2591

```
  logicalModel CDATA #IMPLIED
```

2592

```
  version CDATA #IMPLIED>
```

2593

2594

Definition:

2595

A DocumentSpecification is a collection of Document Definitions. The DocumentSpecification is usually external to the process specification, and is referenced with a URI. An additional reference is to where the logical model is for the Documents in the DocumentSpecification. Typically this would be an ebXML core component context model.

2596

2597

2598

2599

2600

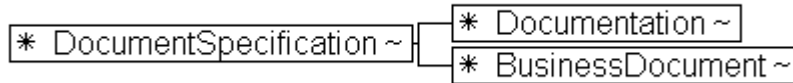
Parents:

2601

- Package

2602

Hierarchical Model:



2603

2604

2605

Attributes:

Attribute Name	Definition	Default Value
name	Defines the generic name of the DocumentSpecification as it is known within this Business Process Specification	Required
nameID	XML ID version of name	Optional
location	Reference to an external source of the schema definition. In the XML Schema version the data type is xsd:anyURI	No default value.
logicalModel	Reference is to where the logical model is. In the XML Schema version the data type is xsd:anyURI	No default value.
version	The version of the document specification	Optional

2606

2607

2608

2609

k. Failure

2610

Element Name: Failure

2611

DTD Declaration:

2612

```
<!ELEMENT Failure (Documentation*) >
```

2613

```
<!ATTLIST Failure
```

2614

```
  fromBusinessState CDATA #REQUIRED
```

2615

```
  fromBusinessStateIDRef IDREF #IMPLIED
```

2616

```
  guardCondition (Success | BusinessFailure |
```

2617

```
    TechnicalFailure | AnyFailure) #IMPLIED
```

2618

```
  guardExpression CDATA #IMPLIED>
```

2619

2620

Definition:

2621

Defines the unsuccessful conclusion of a binary collaboration as a transition from an activity.

2622

2623

Parents:

2624

- BinaryCollaboration

2625

Attributes:

Attribute Name	Definition	Default Value
fromBusinessState	The name of the activity from which this indicates a transition to unsuccessful conclusion of the BusinessTransaction or BinaryCollaboration	Required Input.
fromBusinessState IDRef	The XML IDREF version of fromBusinessState	Optional
guardCondition	The condition that guards this transition	Optional Valid Values: {Success, BusinessFailure, TechnicalFailure, AnyFailure}
guardExpression	An expression whose evaluation results in TRUE or FALSE to determine if this transition should happen or not. The expression can refer to the name or content of the the most recent DocumentEnvelope or content of documents within it.	Optional

2626

2627

I. Fork

2628

Element Name: Fork

2629

DTD Declaration:

2630

```
<!ELEMENT Fork (Documentation*) >
```

2631

```
<!ATTLIST Fork
```

2632

```
    name CDATA #REQUIRED
```

2633

```
    nameID ID #IMPLIED >
```

2634

Definition:

2635

A Fork is a state with one inbound transition and multiple outbound transitions.

2636

All activities pointed to by the outbound transitions are assumed to happen in

2637

parallel.

2638

Parents:

2639

- BinaryCollaboration

2640

Attributes:

2641

Attribute Name	Definition	Default Value
name	Defines the name of the Fork state	Required Input
nameID	The XML ID version of name	Optional

2642

2643

m. Include

2644

Element Name: Include

2645

DTD Declaration:

2646

```
<!ELEMENT Include (Documentation*) >
```

2647

```
<!ATTLIST Include
```

2648

```
  name      CDATA      #REQUIRED
```

2649

```
  version   CDATA      #REQUIRED
```

2650

```
  uuid      CDATA      #REQUIRED
```

2651

```
  uri       CDATA      #REQUIRED >
```

2652

Definition:

2653

Includes another process specification document and merges that specification with the current specification. Any elements of the same name and in the same name scope must have exactly the same specification except that packages may have additional content.

2654

2655

2656

2657

Documents are merged based on name scope. A name in an included package will be indistinguishable from a name in the base document.

2658

2659

Parents:

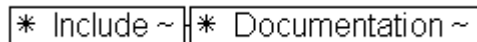
2660

- ProcessSpecification

2661

Hierarchical Model:

2662



2663

Attributes:

Attribute Name	Definition	Default Value
name	Defines the name of a model element. This name must be unique within the context of the model element and will be used to reference the element from other points in the model.	Required
uri	Uniform Resource Indicator. In the XML Schema data type is <code>xsd:anyURI</code>	Required
uuid	Universally unique identifier.	Required
version	Version of the included specification.	Required

2664

2665

2666

n. Join

2667

Element Name: Join

2668

DTD Declaration:

2669

```
<!ELEMENT Join (Documentation*) >
```

2670

```
<!ATTLIST Join
```

2671

```
  name      CDATA      #REQUIRED
```

2672

```
  nameID    ID          #IMPLIED
```

2673

```
  waitAll   (true | false) "true">
```

2674

Definition:

2675 A business state where an activity is waiting for the completion of one or more
 2676 other activities. Defines the point where previously forked activities join up again.

2677

Parents:

2678

- BinaryCollaboration

2679

Attributes:

Attribute Name	Definition	Default Value
name	Defines the name of the Join state.	Required Input
nameID	The XML ID version of name	Optional
waitForAll	Boolean value indicating if this Join state should wait for all incoming transitions to complete. If TRUE, wait for all, if False proceed on first incoming transition.	true Valid Values: {true, false}

2680

2681

o. MultiParty Collaboration

2682

Element Name: MultiPartyCollaboration

2683

DTD Declaration:

2684

```
<!ELEMENT MultiPartyCollaboration (Documentation*,
2685                                     BusinessPartnerRole+) >
```

2686

```
<!ATTLIST MultiPartyCollaboration
2687     name      CDATA #REQUIRED
2688     nameID    ID      #IMPLIED >
```

2687

2688

2689

Definition:

2690

A Multiparty Collaboration is a synthesis of Binary Collaborations. A Multiparty Collaboration consists of a number of Business Partner Roles each playing roles in binary collaborations with each other.

2691

2692

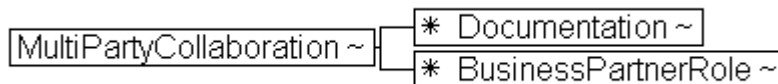
2693

Parents:

2694

- Package

2695

Hierarchical Model:

2696

2697

Attributes:

Attribute Name	Definition	Default Value
name	Defines the name of the MultiPartyCollaboration	Required Input
nameID	The XML ID version of name	Optional Input

2698

2699

2700

p. Package

2701

Element Name: Package

2702

DTD Declaration:

2703

```
<!ELEMENT Package (Documentation*, (Package |
2704 BinaryCollaboration |
2705 MultiPartyCollaboration |
2706 BusinessTransaction)*) >
```

2707

```
<!ATTLIST Package
2708 name CDATA #REQUIRED
2709 nameID ID #IMPLIED >
```

2710

Definition:

2711

Defines a hierarchical name scope containing reusable elements.

2712

Parents:

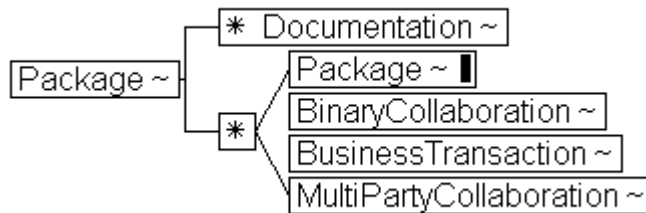
2713

- ProcessSpecification
- Package

2714

2715

2716

Hierarchical Model:

2717

2718

2719

Attributes:

Attribute Name	Definition	Default Value
name	Defines the name of a model element. This name must be unique within the context of the model element and will be used to reference the element from other points in the model.	Required Input
nameID	XML ID version of name	Optional

2720

2721

2722

q. Performs

2723

Element Name: Performs

2724

DTD Declaration:

2725

```
<!ELEMENT Performs (Documentation*) >
```

2726

```
<!ATTLIST Performs
2727 authorizedRole CDATA #REQUIRED
2728 authorizedRoleIDRef IDREF #IMPLIED >
```

2729

Definition:

2730 Performs is an explicit modeling of the relationship between a
 2731 BusinessPartnerRole and the Roles it plays. This specifies the use of an
 2732 Authorized Role within a multiparty collaboration.

2733 **Parents:**

- 2734 • BusinessPartnerRole

2735 **Attributes:**

Attribute Name	Definition	Default Value
authorizedRole	The AuthorizedRole that will be performed by the Business PartnerRole, qualified with the name of the BinaryCollaboration	Required Input
authorizedRoleIDRef	The XML IDREF version of AuthorizedRole	Optional Input

2736

2737

2738

2739 r. **ProcessSpecification**

2740 **Element Name:** ProcessSpecification

2741 **DTD Declaration:**

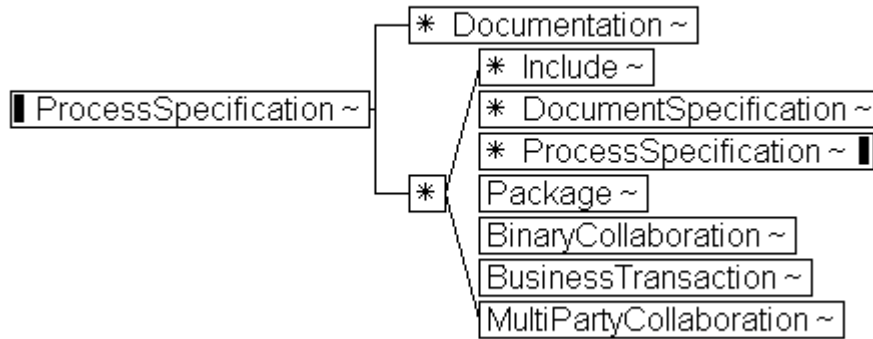
```

2742 <!ELEMENT ProcessSpecification (Documentation*, (Include* |
2743 DocumentSpecification* | ProcessSpecification* | Package |
2744 BinaryCollaboration | BusinessTransaction |
2745 MultiPartyCollaboration)*)>
2746 <!ATTLIST ProcessSpecification
2747     name          ID          #REQUIRED
2748     version       CDATA       #REQUIRED
2749     uuid          CDATA       #REQUIRED >
  
```

2750 **Definition:**

2751 Root element of a process specification document that has a globally unique
 2752 identity.

2753 **Hierarchical Model:**



2754
2755

Attributes:

Attribute Name	Definition	Default Value
name	Defines the name of a model element. This name must be unique within the context of the model element and will be used to reference the element from other points in the model. It is defined as an XML ID.	Required
uuid	Universally unique identifier.	Required
version	Version of the specification.	Required

2756
2757

s. Requesting Business Activity

Element Name: RequestingBusinessActivity

DTD Declaration:

```

2761 <!ELEMENT RequestingBusinessActivity (Documentation*,
2762 DocumentEnvelope) >
2763 <!ATTLIST RequestingBusinessActivity
2764 name CDATA #IMPLIED
2765 nameID ID #IMPLIED
2766 isAuthorizationRequired (true | false) "false"
2767 isIntelligibleCheckRequired (true | false) "false"
2768 isNonRepudiationReceiptRequired (true | false) "false"
2769 isNonRepudiationRequired (true | false) "false"
2770 timeToAcknowledgeAcceptance CDATA #IMPLIED
2771 timeToAcknowledgeReceipt CDATA #IMPLIED>
2772
    
```

Definition:

A RequestingBusinessActivity is a Business Action that is performed by the requesting role within a Business Transaction. It specifies the Document Envelope which will carry the request.

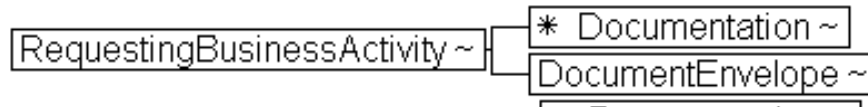
2773
2774
2775

Parents:

- BusinessTransaction

2776
2777

2778

Hierarchical Model:

2779

2780

Attributes:

Attribute Name	Definition	Default Value
name	Defines the name of the RequestingBusinessTransaction	Optional Input
nameID	The XML ID version of name	Optional Input
isAuthorizationRequired	Receiving party must validate identity of originator against a list of authorized originators. This parameter is specified on the sending side. (See also section on action security)	false Valid Values: {true, false}
isIntelligibleCheckRequired	Receiving party must check that a requesting document is not garbled (unreadable, unintelligible) before sending acknowledgement of receipt This parameter is specified on the sending side. (See also section on core transaction semantics)	false Valid Values: {true, false}
isNonRepudiationReceiptRequired	Requires the receiving party to return a signed receipt, and the original sender to save copy of the receipt. This parameter is specified on the sending side. (See also section on nonrepudiation)	false Valid Values: {true, false}
isNonRepudiationRequired	Requires the sending parties to save copies of the transacted documents before sending them (See also section on nonrepudiation)	false Valid Values: {true, false}

timeToAcknowledgeAcceptance	The time a responding role has to non-substantively acknowledge business acceptance of a business document. This parameter is specified on the requesting side. (See also section on core transaction semantics)	No default value.
timeToAcknowledgeReceipt	The time the receiving party has to acknowledge receipt of a business document. This parameter is specified on the sending side. (See also section on core transaction semantics)	No default value.

2781

2782

2783

2784

t. Responding Business Activity

2785

Element Name: RespondingBusinessActivity

2786

DTD Declaration:

2787

```
<!ELEMENT RespondingBusinessActivity (Documentation*,
                                     DocumentEnvelope) >
```

2788

2789

```
<!ATTLIST RespondingBusinessActivity
```

2790

```
  name                CDATA                #IMPLIED
```

2791

```
  nameID              ID                    #IMPLIED
```

2792

```
  isAuthorizationRequired (true | false) "false"
```

2793

```
  isIntelligibleCheckRequired (true | false) "false"
```

2794

```
  isNonRepudiationReceiptRequired (true | false) "false"
```

2795

```
  isNonRepudiationRequired (true | false) "false"
```

2796

```
  timeToAcknowledgeReceipt CDATA                #IMPLIED>
```

2797

Definition:

2798

A RespondingBusinessActivity is a Business Action that is performed by the responding role within a Business Transaction. It specifies the Document Envelope which will carry the response.

2799

2800

2801

There may be multiple possible response Document Envelopes defined, but only one of them will be sent during an actual transaction instance.

2802

2803

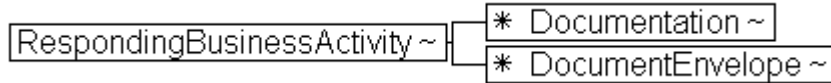
Parents:

2804

- BusinessTransaction

2805

Hierarchical Model:



2806

2807

Attributes:

Attribute Name	Definition	Default Value
name	Defines the name of the RespondingBusinessTransaction	Optional Input
nameID	The XML ID version of name	Optional Input
isAuthorizationRequired	Receiving party must validate identity of originator against a list of authorized originators. This parameter is specified on the sending side. (See also section on action security)	false Valid Values: {true, false}
isIntelligibleCheckRequired	Receiving party must check that a requesting document is not garbled (unreadable, unintelligible) before sending acknowledgement of receipt This parameter is specified on the sending side. (See also section on core transaction semantics)	false Valid Values: {true, false}
isNonRepudiationReceiptRequired	Requires the receiving party to return a signed receipt, and the original sender to save copy of the receipt. This parameter is specified on the sending side. (See also section on nonrepudiation)	false Valid Values: {true, false}
isNonRepudiationRequired	Requires the sending parties to save copies of the transacted documents before sending them (See also section on nonrepudiation)	false Valid Values: {true, false}
timeToAcknowledgeReceipt	The time the receiving party has to acknowledge receipt	No default

t	of a business document. This parameter is specified on the sending side. (See also section on core transaction semantics)	value.
---	---	--------

2808

2809

2810

2811

u. Start

2812

Element Name: Start

2813

DTD Declaration:

2814

```
<!ELEMENT Start (Documentation*) >
```

2815

```
<!ATTLIST Start
```

2816

```
toBusinessState CDATA #REQUIRED
```

2817

```
toBusinessStateIDRef IDREF #IMPLIED >
```

2818

Definition:

2819

The starting state for an Binary Collaboration. A Binary Collaboration should have at least one starting activity. If none defined, then all activities are considered allowable entry points.

2820

2821

2822

Parents:

2823

- BinaryCollaboration

2824

Attributes:

Attribute Name	Definition	Default Value
toBusinessState	The name of an activity which an allowable starting point for this for BinaryCollaboration	Required Input
toBusinessStateIDRef	The XML IDREF version of toBusinessState	Optional

2825

2826

2827

v. Success

2828

Element Name: Success

2829

DTD Declaration:

2830

```
<!ELEMENT Success (Documentation*) >
```

2831

```
<!ATTLIST Success
```

2832

```
fromBusinessState CDATA #REQUIRED
```

2833

```
guardCondition (Success | BusinessFailure |
```

2834

```
TechnicalFailure | AnyFailure) #IMPLIED
```

2835

```
guardExpression CDATA #IMPLIED>
```

2836

2837

Definition:

2838 Defines the successful conclusion of a binary collaboration as a transition from
2839 an activity.

2840 **Parents:**

- BinaryCollaboration

2842 **Attributes:**

Attribute Name	Definition	Default Value
fromBusinessState	The name of the activity from which this indicates a transition to successful conclusion of the BusinessTransaction or BinaryCollaboration	Required Input.
guardCondition	The condition that guards this transition	Optional Valid Values: {Success, BusinessFailure, TechnicalFailure, AnyFailure}
guardExpression	An expression whose evaluation results in TRUE or FALSE as the determination of whether this DocumentEnvelope should be considered the successful initiation or conclusion of a BusinessTransaction. In some situations this could be an XPath expression that interrogates the BusinessDocument.	Optional

2843

2844 **w. Transition**

2845 **ELEMENT Name:** Transition

2846 **DTD Declaration:**

```
2847 <!ELEMENT Transition (Documentation*) >
2848 <!ATTLIST Transition
2849     onInitiation (true | false) "false"
2850     fromBusinessState CDATA #IMPLIED
2851     fromBusinessStateIDRef IDREF #IMPLIED
2852     toBusinessState CDATA #IMPLIED
2853     toBusinessStateIDRef IDREF #IMPLIED
2854     guardCondition (Success | BusinessFailure |
2855     TechnicalFailure | AnyFailure) #IMPLIED
2856     guardExpression CDATA #IMPLIED>
```

2857 **Definition:**

2858 A transition is a transition between two business states in a binary collaboration.

2859 Choreography is expressed as transitions between business states.

2860 **Parents:**

2861

- BinaryCollaboration

2862

- BusinessPartnerRole

2863

Attributes:

Attribute Name	Definition	Default Value
onInitiation	This specifies this is a nested BusinessTransactionActivity and that upon receipt of the request in the associated transaction a second activity is performed before returning to the transaction to send the response back to the original requestor	false Valid Values: {true, false}
fromBusinessState	The name of the state transitioned from	No default value.
fromBusinessStateIDRef	The XML IDREF version of fromBusinessState	Optional
toBusinessState	The name of the state transitioned to	No default value.
toBusinessStateIDRef	The XML IDREF version of toBusinessState	Optional
guardCondition	A reference to the status of the previous transaction. A fixed value of Success, BusinessFailure, TechnicalFailure, or AnyFailure	Optional Valid Values: {Success, BusinessFailure, TechnicalFailure, AnyFailure}
guardExpression	An expression whose evaluation results in TRUE or FALSE to determine if this transition should happen or not. The expression can refer to the name or content of the the most recent DocumentEnvelope or content of documents within it. In some situations this could be an XPath expression that interrogates the BusinessDocument.	Optional

2864

2865

2866

2867

2868

2869

2870

2871 **8.2 XML to UML cross-reference**

2872

2873

2874

2875

The following is a table that references the XML element names in the DTD to their counterpart classes in the UML specification schema.

XML Element	UML Class
Attachment	Attachment
AuthorizedRole	AuthorizedRole
Binary Collaboration	Binary Collaboration
BusinessPartner Role	BusinessPartner Role
Business Transaction Activity	Business Transaction Activity
Business Transaction	Business Transaction
Responding BusinessActivity	Responding BusinessActivity
Requesting BusinessActivity	Requesting BusinessActivity
Collaboration Activity	Collaboration Activity
DocumentEnvelope	DocumentEnvelope
Documentation	None (Should be added)
ebXML Process Specification	(From Package model: ebXML Process Specification)
Failure	Failure
Include	(From Package model: Include)

MultiParty Collaboration	MultiParty Collaboration
Package	(From Package model: Package)
Performs	Performs
Schema	Schema
Fork	Fork
Start	Start
Success	Success
Join	Join
Transition	Transition

2876

2877

2878

2879

The following classes in the UML specification schema are abstract, and do not have an element equivalent in the DTD. Only their concrete subtypes are in the DTD:

2880

- BusinessState

2881

- CompletionState

2882

- BusinessActivity

2883

- BusinessAction

2884

- DocumentSecurity

2885

The following classes in the UML specification schema are in the DTD represented not by elements but by attributes in other elements:

2886

2887

- BusinessDocument (attribute of DocumentEnvelope and of Schema)

2888

2889

2890

2891

2892

8.3 Scoped Name Reference

2893

The structure of ebXML process specifications encourages re-use. An ebXMLProcessSpecification can include another ebXMLProcessSpecification by reference.

2894

2895

2896

In addition the contents of a ebXMLProcessSpecification can be arranged in a recursive package structure. The ebXMLProcessSpecification is a package container, so it can contain packages within it. Package in itself is also a package container, so it can contain further packages within it.

2897

2898

2899

2900

Packages function as namespaces as per below.

2901 Finally a Package, at any level can have PackageContent. Types of
 2902 PackageContent are BusinessTransaction, BinaryCollaboration,
 2903 MultiPartyCollaboration.

2904 PackageContent are always uniquely named within a package. Lower level
 2905 elements a uniquely named within their parent PackageContent.

2906 Each PackageContent type is a built-in context provider for the core components
 2907 Logical Model for the Business Document definitions referenced by this
 2908 ebXMLProcessSpecification.

2909 Within a ebXMLProcessSpecification the following applies to naming:

2910 Specification elements reference other specification elements by name through
 2911 the use of attributes. The design pattern is that elements have a name attribute
 2912 and other elements that reference the named elements do so through an
 2913 attribute defined as the lowerCamelCase version of the referenced element (e.g.
 2914 AuthorizedRole has attribute name while Performs, which references
 2915 AuthorizedRole, has attribute authorizedRole). Two types of attributes are
 2916 provided for names and references, XML ID/IDREF based and plain text. Each
 2917 named element has a required name attribute and an optional nameID attribute.
 2918 Referencing elements have lowerCamelCase and lowerCamelCaseIDRef
 2919 attributes for the referenced element. XML ID/IDREF functionality requires all IDs
 2920 to be unique within a document and that all IDREFs point to a defined ID value.
 2921 Plain text attributes do not have this capability and may result is duplicate names.
 2922 To unambiguously identify a referenced element using plain text attribute in the
 2923 referencing attribute it is strongly recommended that XPath syntax be used.
 2924 However, this is not enforced in the DTD or Schema.

2925 The purpose of providing both solutions is to facilitate creation of Process
 2926 Specification Documents directly in XML and to support future development tools
 2927 that can automatically assign machine readable nameIDs and references. Both
 2928 styles can be used simultaneously, in which case the ID and IDREF versions
 2929 provide the unambiguous referencing and the plain text versions are used to
 2930 provide meaningful names. Examples of named elements and references:

2931 <Package name="ebXMLOrdering">
 2932 <BinaryCollaboration name="OrderCollaboration" nameID="b112">
 2933 <AuthorizedRole name="buyer" nameID="r224"/>
 2934 <AuthorizedRole name="seller" nameID="r225"/>
 2935 </BinaryCollaboration>
 2936 </Package>

2937

2938 <!--the XPath approach -->
 2939 <Performs
 2940 authorizedRole='//Package[@name="OAGOrdering"]/BinaryCollaboration[@name="OrderColla
 2941 boration"]/AuthorizedRole[@name="buyer"]'/>
 2942

2943 <!--Combination approach -->
 2944 <Performs authorizedRole="buyer" authorizedRoleIDRef="r222"/>
 2945

2946 It is not required to use the full path specification as shown above, other
 2947 forms of XPath expressions could be used as long as they resolve to a single

2948 reference. For example if buyer was unique to the document then the XPath
 2949 could have been:
 2950 <Performs authorizedRole='//AuthorizedRole[@name="buyer"]'/>
 2951 Relative paths are also allowed for example:
 2952 <BusinessTransactionActivity fromAuthorizedRole='../AuthorizedRole[@name="buyer"]' ... />

2953 **8.4 Sample XML document against above DTD**

2954
 2955 Provided in Appendix A
 2956

2957 **9 Business signal structures**

2958
 2959 The ebXML Message Service Specification signal structures provide
 2960 business service state alignment infrastructure, including unique message
 2961 identifiers and digests used to meet the basic process alignment
 2962 requirements. The business signal payload structures provided herein
 2963 are optional and normative and are intended to provide business and
 2964 legal semantic to the business signals. Since signals do not differ in
 2965 structure from business transaction to business transaction, they are
 2966 defined once and for all, and their definition is implied by the conjunction
 2967 of the Business Process Specification Schema and Message Service
 2968 Specification. Here are the DTD's for business signal payload for
 2969 receiptAcknowledgment (from the RosettaNet website, courtesy of
 2970 RosettaNet, and Edifecs) and for acceptanceAcknowledgement and
 2971 exception.

2972 **9.1.1 ReceiptAcknowledgment DTD**

```

2973 <!--
2974 RosettaNet XML Message Schema.
2975 http://www.rosettnet.org
2976 RosettaNet XML Message Schema.
2977 Receipt Acknowledgement
2978 Version 1.1
2979 -->
2980
2981 <!ENTITY % common-attributes "id CDATA #IMPLIED">
2982
2983 <!ELEMENT ReceiptAcknowledgement (
2984   fromRole ,
2985   NonRepudiationInformation? ,
2986   receivedDocumentDateTime ,
2987   receivedDocumentIdentifier ,
2988   thisMessageDateTime ,
2989   thisMessageIdentifier ,
2990
2991
```

```
2992         toRole ) >
2993
2994     <!ELEMENT fromRole
2995         ( PartnerRoleDescription ) >
2996
2997     <!ELEMENT PartnerRoleDescription (
2998         ContactInformation? ,
2999         GlobalPartnerRoleClassificationCode ,
3000         PartnerDescription ) >
3001
3002     <!ELEMENT ContactInformation (
3003         contactName ,
3004         EmailAddress ,
3005         telephoneNumber ) >
3006
3007     <!ELEMENT contactName
3008         ( FreeFormText ) >
3009
3010     <!ELEMENT FreeFormText
3011         ( #PCDATA ) >
3012     <!ATTLIST FreeFormText
3013         xml:lang CDATA #IMPLIED >
3014
3015     <!ELEMENT EmailAddress
3016         ( #PCDATA ) >
3017
3018     <!ELEMENT telephoneNumber
3019         ( CommunicationsNumber ) >
3020
3021     <!ELEMENT CommunicationsNumber
3022         ( #PCDATA ) >
3023
3024     <!ELEMENT GlobalPartnerRoleClassificationCode
3025         ( #PCDATA ) >
3026
3027     <!ELEMENT PartnerDescription (
3028         BusinessDescription ,
3029         GlobalPartnerClassificationCode ) >
3030
3031     <!ELEMENT BusinessDescription (
3032         GlobalBusinessIdentifier ,
3033         GlobalSupplyChainCode ) >
3034
3035     <!ELEMENT GlobalBusinessIdentifier
3036         ( #PCDATA ) >
3037
3038     <!ELEMENT GlobalSupplyChainCode
3039         ( #PCDATA ) >
3040
3041     <!ELEMENT GlobalPartnerClassificationCode
3042         ( #PCDATA ) >
3043
3044     <!ELEMENT NonRepudiationInformation (
3045         GlobalDigestAlgorithmCode ,
3046         OriginalMessageDigest ) >
```

```

3047
3048      <!ELEMENT GlobalDigestAlgorithmCode
3049          ( #PCDATA ) >
3050
3051      <!ELEMENT OriginalMessageDigest
3052          ( #PCDATA ) >
3053
3054      <!ELEMENT receivedDocumentDateTime
3055          ( DateTimeStamp ) >
3056
3057      <!ELEMENT DateTimeStamp
3058          ( #PCDATA ) >
3059
3060      <!ELEMENT receivedDocumentIdentifier
3061          ( ProprietaryDocumentIdentifier ) >
3062
3063      <!ELEMENT ProprietaryDocumentIdentifier
3064          ( #PCDATA ) >
3065
3066      <!ELEMENT thisMessageDateTime
3067          ( DateTimeStamp ) >
3068
3069      <!ELEMENT thisMessageIdentifier
3070          ( ProprietaryMessageIdentifier ) >
3071
3072      <!ELEMENT ProprietaryMessageIdentifier
3073          ( #PCDATA ) >
3074
3075      <!ELEMENT toRole
3076          ( PartnerRoleDescription ) >
3077

```

9.1.2 AcceptanceAcknowledgement DTD

```

3078      <!--
3079          RosettaNet XML Message Schema.
3080          http://www.rosettanet.org
3081          RosettaNet XML Message Schema.
3082          Acceptance Acknowledgement Exception
3083          Version 1.1
3084      -->
3085
3086
3087      <!ENTITY % common-attributes "id CDATA #IMPLIED">
3088
3089      <!ELEMENT AcceptanceAcknowledgementException (
3090          fromRole ,
3091          reason ,
3092          theMessageDatetime ,
3093          theOffendingDocumentDateTime ,
3094          theOffendingDocumentIdentifier ,
3095          thisMessageIdentifier ,
3096          toRole ) >
3097
3098      <!ELEMENT fromRole
3099          ( PartnerRoleDescription ) >
3100

```

```
3101      <!ELEMENT PartnerRoleDescription (
3102          ContactInformation? ,
3103          GlobalPartnerRoleClassificationCode ,
3104          PartnerDescription ) >
3105
3106      <!ELEMENT ContactInformation (
3107          contactName ,
3108          EmailAddress ,
3109          telephoneNumber ) >
3110
3111      <!ELEMENT contactName
3112          ( FreeFormText ) >
3113
3114      <!ELEMENT FreeFormText
3115          ( #PCDATA ) >
3116      <!ATTLIST FreeFormText
3117          xml:lang CDATA #IMPLIED >
3118
3119      <!ELEMENT EmailAddress
3120          ( #PCDATA ) >
3121
3122      <!ELEMENT telephoneNumber
3123          ( CommunicationsNumber ) >
3124
3125      <!ELEMENT CommunicationsNumber
3126          ( #PCDATA ) >
3127
3128      <!ELEMENT GlobalPartnerRoleClassificationCode
3129          ( #PCDATA ) >
3130
3131      <!ELEMENT PartnerDescription (
3132          BusinessDescription ,
3133          GlobalPartnerClassificationCode ) >
3134
3135      <!ELEMENT BusinessDescription (
3136          GlobalBusinessIdentifier ,
3137          GlobalSupplyChainCode ) >
3138
3139      <!ELEMENT GlobalBusinessIdentifier
3140          ( #PCDATA ) >
3141
3142      <!ELEMENT GlobalSupplyChainCode
3143          ( #PCDATA ) >
3144
3145      <!ELEMENT GlobalPartnerClassificationCode
3146          ( #PCDATA ) >
3147
3148      <!ELEMENT reason
3149          ( FreeFormText ) >
3150
3151      <!ELEMENT theMessageDatetime
3152          ( DateTimeStamp ) >
3153
3154      <!ELEMENT DateTimeStamp
3155          ( #PCDATA ) >
```

```

3156
3157      <!ELEMENT theOffendingDocumentDateTime
3158          ( DateTimeStamp ) >
3159
3160      <!ELEMENT theOffendingDocumentIdentifier
3161          ( ProprietaryDocumentIdentifier ) >
3162
3163      <!ELEMENT ProprietaryDocumentIdentifier
3164          ( #PCDATA ) >
3165
3166      <!ELEMENT thisMessageIdentifier
3167          ( ProprietaryMessageIdentifier ) >
3168
3169      <!ELEMENT ProprietaryMessageIdentifier
3170          ( #PCDATA ) >
3171
3172      <!ELEMENT toRole
3173          ( PartnerRoleDescription ) >
3174

```

3175 9.1.3 Exception Signal DTD

```

3176
3177      <!--
3178          RosettaNet XML Message Schema.
3179          http://www.rosettanet.org
3180          RosettaNet XML Message Schema.
3181          Exception
3182          Version 1.1
3183      -->
3184
3185
3186      <!ENTITY % common-attributes "id CDATA #IMPLIED">
3187
3188      <!ELEMENT Exception (
3189          fromRole? ,
3190          reason ,
3191          theMessageDatetime ,
3192          theOffendingDocumentDateTime? ,
3193          theOffendingDocumentIdentifier? ,
3194          thisMessageIdentifier ,
3195          toRole? ) >
3196
3197      <!ELEMENT fromRole
3198          ( PartnerRoleDescription ) >
3199
3200      <!ELEMENT PartnerRoleDescription (
3201          ContactInformation? ,
3202          GlobalPartnerRoleClassificationCode? ,
3203          PartnerDescription? ) >
3204
3205      <!ELEMENT ContactInformation (
3206          contactName? ,
3207          EmailAddress? ,
3208          telephoneNumber? ) >
3209

```



```
3210      <!ELEMENT contactName
3211          ( FreeFormText ) >
3212
3213      <!ELEMENT FreeFormText
3214          ( #PCDATA ) >
3215      <!ATTLIST FreeFormText
3216          xml:lang CDATA #IMPLIED >
3217
3218      <!ELEMENT EmailAddress
3219          ( #PCDATA ) >
3220
3221      <!ELEMENT telephoneNumber
3222          ( CommunicationsNumber ) >
3223
3224      <!ELEMENT CommunicationsNumber
3225          ( #PCDATA ) >
3226
3227      <!ELEMENT GlobalPartnerRoleClassificationCode
3228          ( #PCDATA ) >
3229
3230      <!ELEMENT PartnerDescription (
3231          BusinessDescription? ,
3232          GlobalPartnerClassificationCode? ) >
3233
3234      <!ELEMENT BusinessDescription (
3235          GlobalBusinessIdentifier? ,
3236          GlobalSupplyChainCode? ) >
3237
3238      <!ELEMENT GlobalBusinessIdentifier
3239          ( #PCDATA ) >
3240
3241      <!ELEMENT GlobalSupplyChainCode
3242          ( #PCDATA ) >
3243
3244      <!ELEMENT GlobalPartnerClassificationCode
3245          ( #PCDATA ) >
3246
3247      <!ELEMENT reason
3248          ( FreeFormText ) >
3249
3250      <!ELEMENT theMessageDatetime
3251          ( DateTimeStamp ) >
3252
3253      <!ELEMENT DateTimeStamp
3254          ( #PCDATA ) >
3255
3256      <!ELEMENT theOffendingDocumentDateTime
3257          ( DateTimeStamp ) >
3258
3259      <!ELEMENT theOffendingDocumentIdentifier
3260          ( ProprietaryDocumentIdentifier ) >
3261
3262      <!ELEMENT ProprietaryDocumentIdentifier
3263          ( #PCDATA ) >
3264
```

```

3265      <!ELEMENT thisMessageIdentifier
3266          ( ProprietaryMessageIdentifier ) >
3267
3268      <!ELEMENT ProprietaryMessageIdentifier
3269          ( #PCDATA ) >
3270
3271      <!ELEMENT toRole
3272          ( PartnerRoleDescription ) >
3273
3274
3275

```

3276 10 Production Rules

3277 This section provides a set of production rules, defining the mapping from the
 3278 UML version of the *Business Process Specification Schema* to the XML version.

3279 The primary purpose for these production rules is to govern the one-time
 3280 generation of the DTD version of the *Business Process Specification Schema*
 3281 from the UML Class Diagram version of *Business Process Specification Schema*.

3282 The Class Diagram version of *Business Process Specification Schema* is not
 3283 intended for the direct creation of ebXML Business Process Specifications.
 3284 However, if a *Business Process Specification* was in fact (programmatically)
 3285 created as an instance of this class diagram, the production rules would also
 3286 provide the prescriptive definition necessary to translate a such an instance into
 3287 a XML Specification Document conformant with the DTD. The production rules
 3288 are defined for concrete classes, abstract classes, aggregate associations,
 3289 specialization associations and unidirectional associations.

- 3290 1. Classes are rendered as XML elements.
- 3291 2. Class attributes are rendered as XML attributes. NOTE: occurrence
 3292 requirements (required vs optional) and default values for attributes are not
 3293 modeled.
- 3294 3. Specialization classes (classes that inherit from another class) are rendered
 3295 as XML elements including all attributes and aggregate associations from the
 3296 base class. Repeated attributes are normalized to a single occurrence.
- 3297 4. Abstract classes are not rendered in the XML DTD. Abstract classes are
 3298 inherited from and represent a form of collection. A class that aggregates an
 3299 abstract class, essentially aggregates "any of each" of the specialization
 3300 classes.
- 3301 5. An aggregate association renders the aggregated class as an XML child
 3302 element with appropriate cardinality.
- 3303 6. A unidirectional association defines an attribute in the originating class of the
 3304 same name as the class the association points to. This type of attribute is
 3305 called a "reference attribute" and contains the name of the class it points to.
 3306 The referenced class must have a "name" attribute.

- 3307
3308
3309
7. A class attribute data type, that has a class of the same name with stereotype <<Enumeration>> is rendered as an XML attribute enumeration. The Enumeration class does not have an explicit association.
- 3310
3311
3312
8. A class attribute data type (e.g. Time, URI, Boolean) that has no corresponding class definition is rendered as a string in the DTD. In the XML Schema version these data types are mapped as:
- 3313 Time - xsd:duration
3314 URI - xsd:anyURI
3315 Boolean - xsd:boolean
- 3316
3317
3318
9. Each class is given an optional "Documentation*" element which is intended for annotation of the specification instances. This is not modeled.

3319 Appendix A: Sample XML Business Process 3320 Specification

```
3321
3322 <!-- edited with XML Spy v3.5 NT (http://www.xmlspy.com) by Kurt
3323 Kanaskie (Lucent Technologies) -->
3324 <!-- Notes from Kurt Kanaskie on 2001-04-17
3325 Differences from 099 DTD:
3326     binaryCollaboration attribute in Performs
3327     fromBinaryCollaboration attribute in Transition
3328     toBinaryCollaboration attribute in Transition
3329 Differences from original:
3330     guard instead of guardExpression
3331     See EBXMLSpecification_2001_04_23.dtd for other changes
3332 -->
3333 <!DOCTYPE ProcessSpecification SYSTEM "ebXMLProcessSpecification-
3334 v1.00.dtd">
3335 <ProcessSpecification name="Simple" version="1.1" uuid="[1234-5678-
3336 901234]">
3337     <!-- Business Documents -->
3338     <DocumentSpecification name="EBXML">
3339         <BusinessDocument name="Catalog Request"/>
3340         <BusinessDocument name="Catalog"/>
3341         <BusinessDocument name="Purchase Order"/>
3342         <BusinessDocument name="PO Acknowledgement"/>
3343         <BusinessDocument name="Credit Request"/>
3344         <BusinessDocument name="Credit Confirm"/>
3345         <BusinessDocument name="ASN"/>
3346         <BusinessDocument name="CreditAdvice"/>
3347         <BusinessDocument name="DebitAdvice"/>
3348         <BusinessDocument name="Invoice"/>
3349         <BusinessDocument name="Payment"/>
3350         <BusinessDocument name="Inventory Report Request"/>
3351         <BusinessDocument name="Inventory Report"/>
3352         <BusinessDocument name="Inventory Report"/>
3353     </DocumentSpecification>
3354     <Package name="Ordering">
3355         <!-- First the overall MultiParty Collaboration -->
3356         <MultiPartyCollaboration name="DropShip">
3357             <BusinessPartnerRole name="Customer">
3358                 <Performs authorizedRole="requestor"/>
3359                 <Performs authorizedRole="buyer"/>
3360                 <Transition fromBusinessState="Catalog Request"
3361 toBusinessState="Create Order"/>
3362             </BusinessPartnerRole>
3363             <BusinessPartnerRole name="Retailer">
3364                 <Performs authorizedRole="provider"/>
3365                 <Performs authorizedRole="seller"/>
3366                 <Performs authorizedRole="Creditor"/>
3367                 <Performs authorizedRole="buyer"/>
3368                 <Performs authorizedRole="Payee"/>
3369                 <Performs authorizedRole="Payor"/>
3370                 <Performs authorizedRole="requestor"/>
```

```
3371         <Transition fromBusinessState="Create Order"
3372 toBusinessState="Check Credit"/>
3373         <Transition fromBusinessState="Check Credit"
3374 toBusinessState="Create Order"/>
3375     </BusinessPartnerRole>
3376     <BusinessPartnerRole name="DropShip Vendor">
3377         <Performs authorizedRole="seller"/>
3378         <Performs authorizedRole="payee"/>
3379         <Performs authorizedRole="provider"/>
3380     </BusinessPartnerRole>
3381     <BusinessPartnerRole name="Credit Authority">
3382         <Performs authorizedRole="credit service"/>
3383         <Performs authorizedRole="payor"/>
3384     </BusinessPartnerRole>
3385 </MultiPartyCollaboration>
3386 <!-- Now the Binary Collaborations -->
3387 <BinaryCollaboration name="Request Catalog">
3388     <AuthorizedRole name="requestor"/>
3389     <AuthorizedRole name="provider"/>
3390     <BusinessTransactionActivity name="Catalog Request"
3391 businessTransaction="Catalog Request" fromAuthorizedRole="requestor"
3392 toAuthorizedRole="provider"/>
3393 </BinaryCollaboration>
3394 <BinaryCollaboration name="Firm Order" timeToPerform="P2D">
3395     <Documentation>timeToPerform = Period: 2 days from
3396 start of transaction</Documentation>
3397     <AuthorizedRole name="buyer"/>
3398     <AuthorizedRole name="seller"/>
3399     <BusinessTransactionActivity name="Create Order"
3400 businessTransaction="Create Order" fromAuthorizedRole="buyer"
3401 toAuthorizedRole="seller"/>
3402 </BinaryCollaboration>
3403 <BinaryCollaboration name="Product Fulfillment"
3404 timeToPerform="P5D">
3405     <Documentation>timeToPerform = Period: 5 days from
3406 start of transaction</Documentation>
3407     <AuthorizedRole name="buyer"/>
3408     <AuthorizedRole name="seller"/>
3409     <BusinessTransactionActivity name="Create Order"
3410 businessTransaction="Create Order" fromAuthorizedRole="buyer"
3411 toAuthorizedRole="seller"/>
3412     <BusinessTransactionActivity name="Notify shipment"
3413 businessTransaction="Notify of advance shipment"
3414 fromAuthorizedRole="buyer" toAuthorizedRole="seller"/>
3415         <Start toBusinessState="Create Order"/>
3416         <Transition fromBusinessState="Create Order"
3417 toBusinessState="Notify shipment"/>
3418         <Success fromBusinessState="Notify shipment"
3419 guardExpression="Success"/>
3420         <Failure fromBusinessState="Notify shipment"
3421 guardExpression="BusinessFailure"/>
3422     </BinaryCollaboration>
3423 <BinaryCollaboration name="Inventory Status">
3424     <AuthorizedRole name="requestor"/>
3425     <AuthorizedRole name="provider"/>
```

```
3426         <BusinessTransactionActivity name="Inventory Report
3427 Request" businessTransaction="Inventory Report Request"
3428 fromAuthorizedRole="requestor" toAuthorizedRole="provider"/>
3429         <BusinessTransactionActivity name="Inventory Report"
3430 businessTransaction="Inventory Report" fromAuthorizedRole="provider"
3431 toAuthorizedRole="requestor"/>
3432     </BinaryCollaboration>
3433     <BinaryCollaboration name="Credit Inquiry">
3434         <AuthorizedRole name="creditor"/>
3435         <AuthorizedRole name="credit service"/>
3436         <BusinessTransactionActivity name="Check Credit"
3437 businessTransaction="Check Credit" fromAuthorizedRole="creditor"
3438 toAuthorizedRole="credit service"/>
3439     </BinaryCollaboration>
3440     <BinaryCollaboration name="Credit Payment">
3441         <AuthorizedRole name="payee"/>
3442         <AuthorizedRole name="payor"/>
3443         <BusinessTransactionActivity name="Process Credit
3444 Payment" businessTransaction="Process Credit Payment"
3445 fromAuthorizedRole="payee" toAuthorizedRole="payor"/>
3446     </BinaryCollaboration>
3447     <!-- A compound BinaryCollaboration for illustration
3448 purposes-->
3449     <BinaryCollaboration name="Credit Charge">
3450         <AuthorizedRole name="charger"/>
3451         <AuthorizedRole name="credit service"/>
3452         <CollaborationActivity name="Credit Inquiry"
3453 binaryCollaboration="Credit Inquiry" fromAuthorizedRole="creditor"
3454 toAuthorizedRole="credit service"/>
3455         <CollaborationActivity name="Credit Payment"
3456 binaryCollaboration="Credit Payment" fromAuthorizedRole="payee"
3457 toAuthorizedRole="payor"/>
3458         <Transition fromBusinessState="Credit Inquiry"
3459 toBusinessState="Credit Payment"/>
3460     </BinaryCollaboration>
3461     <BinaryCollaboration name="Fulfillment Payment">
3462         <AuthorizedRole name="payee"/>
3463         <AuthorizedRole name="payor"/>
3464         <BusinessTransactionActivity name="Process Payment"
3465 businessTransaction="Process Payment" fromAuthorizedRole="payee"
3466 toAuthorizedRole="payor"/>
3467     </BinaryCollaboration>
3468     <!-- Here are all the Business Transactions needed -->
3469     <BusinessTransaction name="Catalog Request">
3470         <RequestingBusinessActivity name="">
3471             <DocumentEnvelope isPositiveResponse="true"
3472 businessDocument="Catalog Request"/>
3473         </RequestingBusinessActivity>
3474         <RespondingBusinessActivity name="">
3475             <DocumentEnvelope isPositiveResponse="true"
3476 businessDocument="Catalog"/>
3477         </RespondingBusinessActivity>
3478     </BusinessTransaction>
3479     <BusinessTransaction name="Create Order">
```

```
3480         <RequestingBusinessActivity name=""
3481 isNonRepudiationRequired="true" timeToAcknowledgeReceipt="P2D"
3482 timeToAcknowledgeAcceptance="P3D">
3483         <DocumentEnvelope isPositiveResponse="true"
3484 businessDocument="Purchase Order"/>
3485     </RequestingBusinessActivity>
3486     <RespondingBusinessActivity name=""
3487 isNonRepudiationRequired="true" timeToAcknowledgeReceipt="P5D">
3488         <DocumentEnvelope isPositiveResponse="true"
3489 businessDocument="PO Acknowledgement"/>
3490     </RespondingBusinessActivity>
3491 </BusinessTransaction>
3492 <BusinessTransaction name="Check Credit ">
3493     <RequestingBusinessActivity name="">
3494         <DocumentEnvelope isPositiveResponse="true"
3495 businessDocument="Credit Request"/>
3496     </RequestingBusinessActivity>
3497     <RespondingBusinessActivity name="">
3498         <DocumentEnvelope isPositiveResponse="true"
3499 businessDocument="Credit Confirm"/>
3500     </RespondingBusinessActivity>
3501 </BusinessTransaction>
3502 <BusinessTransaction name="Notify of advance shipment">
3503     <RequestingBusinessActivity name="">
3504         <DocumentEnvelope isPositiveResponse="true"
3505 businessDocument="ASN"/>
3506     </RequestingBusinessActivity>
3507     <RespondingBusinessActivity name=""
3508 timeToAcknowledgeReceipt="P2D"/>
3509 </BusinessTransaction>
3510 <BusinessTransaction name="Process Credit Payment">
3511     <RequestingBusinessActivity name="">
3512         <DocumentEnvelope isPositiveResponse="true"
3513 businessDocument="CreditAdvice"/>
3514     </RequestingBusinessActivity>
3515     <RespondingBusinessActivity name="">
3516         <DocumentEnvelope isPositiveResponse="true"
3517 businessDocument="DebitAdvice"/>
3518     </RespondingBusinessActivity>
3519 </BusinessTransaction>
3520 <BusinessTransaction name="Process Payment">
3521     <RequestingBusinessActivity name="">
3522         <DocumentEnvelope isPositiveResponse="true"
3523 businessDocument="Invoice"/>
3524     </RequestingBusinessActivity>
3525     <RespondingBusinessActivity name="">
3526         <DocumentEnvelope isPositiveResponse="true"
3527 businessDocument="Payment"/>
3528     </RespondingBusinessActivity>
3529 </BusinessTransaction>
3530 <BusinessTransaction name="Request Inventory Report">
3531     <RequestingBusinessActivity name="">
3532         <DocumentEnvelope isPositiveResponse="true"
3533 businessDocument="Inventory Report Request"/>
3534     </RequestingBusinessActivity>
```

```
3535         <RespondingBusinessActivity name="">
3536             <DocumentEnvelope isPositiveResponse="true"
3537 businessDocument="Inventory Report"/>
3538         </RespondingBusinessActivity>
3539     </BusinessTransaction>
3540     <BusinessTransaction name="Inventory Report">
3541         <RequestingBusinessActivity name="">
3542             <DocumentEnvelope isPositiveResponse="true"
3543 businessDocument="Inventory Report"/>
3544         </RequestingBusinessActivity>
3545         <RespondingBusinessActivity name=""/>
3546     </BusinessTransaction>
3547 </Package>
3548 </ProcessSpecification>
```


3549 **Appendix B: Business Process Specification Schema**

3550 **DTD**

```
3551
3552 <!-- ===== --
3553 >
3554 <!-- Editor: Kurt Kanaskie (Lucent Technologies) --
3555 >
3556 <!-- Version: Version 1.0 --
3557 >
3558 <!-- Updated: 2001-04-27 --
3559 >
3560 <!-- --
3561 >
3562 <!-- Public Identifier: --
3563 >
3564 <!--          "-//ebXML//DTD Process Specification ver 1.0//EN" --
3565 >
3566 <!-- --
3567 >
3568 <!-- Purpose: --
3569 >
3570 <!-- The ebXML Specification DTD provides a standard --
3571 >
3572 <!-- framework by which business systems may be --
3573 >
3574 <!-- configured to support execution of business --
3575 >
3576 <!-- transactions. It is based upon prior UN/CEFACT --
3577 >
3578 <!-- work, specifically the metamodel behind the --
3579 >
3580 <!-- UN/CEFACT Unified Modeling Methodology (UMM) defined --
3581 >
3582 <!-- in the N090R9.1 specification. --
3583 -->
3584 <!-- --
3585 >
3586 <!-- The Specification Schema supports the specification --
3587 >
3588 <!-- of Business Transactions and the choreography of --
3589 >
3590 <!-- Business Transactions into Business Collaborations. --
3591 >
3592 <!-- --
3593 >
3594 <!-- Notes: --
3595 >
3596 <!-- time periods are represented using ISO 8601 format --
3597 >
3598 <!-- (e.g. P2D for 2 Days, P2H30M for 2 Hours 30 Minutes --
3599 >
```

```

3600 <!-- --
3601 >
3602 <!-- Naming and reference is based on convention that an --
3603 >
3604 <!-- Element with a name attribute (e.g. AuthorizedRole) --
3605 >
3606 <!-- is refernced by an attribute in another element with --
3607 >
3608 <!-- the name in lowerCamelCase (e.g. authorizedRole). --
3609 >
3610 <!-- --
3611 >
3612 <!-- fromBusinessState and toBusinessState refer to the --
3613 >
3614 <!-- the names of a BusinessTransactionActivity, --
3615 >
3616 <!-- CollaborationActivity, Fork, and Join, all are targets for --
3617 >
3618 <!-- from/to in Transition. This deviates from the normal --
3619 >
3620 <!-- convention of lowerCamelCase attribute name --
3621 >
3622 <!-- BusinessState is used as a generic term for: --
3623 >
3624 <!-- Fork, Join, Success, Failure --
3625 >
3626 <!-- --
3627 >
3628 <!-- Constraints: --
3629 >
3630 <!-- - attributes location, logicalModel, pattern, specification --
3631 >
3632 <!-- uri, are of type xsd:anyURI -->
3633 <!-- - attributes timeTo* are of type xsd:duration --
3634 >
3635 <!-- isSuccess is an expression (e.g. XPath) that results --
3636 >
3637 <!-- in a boolean true or false based on document name or --
3638 >
3639 <!-- document content. --
3640 >
3641 <!-- --
3642 >
3643 <!-- ===== --
3644 >
3645
3646 <!ELEMENT ProcessSpecification (Documentation*, (Include* |
3647 DocumentSpecification* | ProcessSpecification* |
3648 Package | BinaryCollaboration |
3649 BusinessTransaction | MultiPartyCollaboration)*)>
3650 <!ATTLIST ProcessSpecification
3651 name ID #REQUIRED
3652 uuid CDATA #REQUIRED
3653 version CDATA #REQUIRED
3654 >

```

```
3655 <!ELEMENT Documentation (#PCDATA)>
3656 <!ATTLIST Documentation
3657     uri CDATA #IMPLIED
3658 >
3659
3660 <!ELEMENT Include (Documentation*)>
3661 <!ATTLIST Include
3662     name CDATA #REQUIRED
3663     uuid CDATA #REQUIRED
3664     uri CDATA #REQUIRED
3665     version CDATA #REQUIRED
3666 >
3667 <!ELEMENT DocumentSpecification (Documentation*, BusinessDocument*)>
3668 <!ATTLIST DocumentSpecification
3669     name CDATA #REQUIRED
3670     nameID ID #IMPLIED
3671     location CDATA #IMPLIED
3672     logicalModel CDATA #IMPLIED
3673     version CDATA #IMPLIED
3674 >
3675 <!ELEMENT BusinessDocument (Documentation*)>
3676 <!ATTLIST BusinessDocument
3677     name CDATA #REQUIRED
3678     nameID ID #IMPLIED
3679 >
3680 <!ELEMENT Package (Documentation*,
3681     (Package | BinaryCollaboration | BusinessTransaction |
3682     MultiPartyCollaboration)*)>
3683 <!ATTLIST Package
3684     name CDATA #REQUIRED
3685     nameID ID #IMPLIED
3686 >
3687 <!-- Model requires 2 Authorized roles -->
3688 <!ELEMENT BinaryCollaboration (Documentation*, AuthorizedRole,
3689     AuthorizedRole,
3690     (Documentation* | Start | Transition | Success
3691     | Failure |
3692     BusinessTransactionActivity |
3693     CollaborationActivity | Fork | Join)*)>
3694 <!ATTLIST BinaryCollaboration
3695     name CDATA #REQUIRED
3696     nameID ID #IMPLIED
3697     pattern CDATA #IMPLIED
3698     beginsWhen CDATA #IMPLIED
3699     endsWhen CDATA #IMPLIED
3700     preCondition CDATA #IMPLIED
3701     postCondition CDATA #IMPLIED
3702     timeToPerform CDATA #IMPLIED
3703 >
3704 <!ELEMENT MultiPartyCollaboration (Documentation*,
3705     BusinessPartnerRole*)>
3706 <!ATTLIST MultiPartyCollaboration
3707     name CDATA #REQUIRED
3708     nameID ID #IMPLIED
3709 >
```

```
3710
3711 <!ELEMENT AuthorizedRole (Documentation*)>
3712 <!ATTLIST AuthorizedRole
3713     name CDATA #REQUIRED
3714     nameID ID #IMPLIED
3715 >
3716
3717 <!-- A BusinessState is one of Start, Success, Failure, Fork, Join,
3718 BusinessTransactionActivity or CollaborationActivity -->
3719 <!-- fromBusinessState and toBusinessState are fully qualified using
3720 XPath -->
3721 <!-- "guardExpression" is an expression that results in a boolean true
3722 or false -->
3723 <!ELEMENT Transition (Documentation*)>
3724 <!ATTLIST Transition
3725     onInitiation (true | false) "false"
3726     fromBusinessState CDATA #IMPLIED
3727     fromBusinessStateIDRef IDREF #IMPLIED
3728     toBusinessState CDATA #IMPLIED
3729     toBusinessStateIDRef IDREF #IMPLIED
3730     guardCondition (Success | BusinessFailure | TechnicalFailure |
3731 AnyFailure ) #IMPLIED
3732     guardExpression CDATA #IMPLIED
3733 >
3734 <!-- Start is a special type of Transition in that it only has a
3735 destination -->
3736 <!ELEMENT Start (Documentation*)>
3737 <!ATTLIST Start
3738     toBusinessState CDATA #REQUIRED
3739     toBusinessStateIDRef IDREF #IMPLIED
3740 >
3741 <!-- Success is a special type of Transition in that it only has a
3742 origination -->
3743 <!ELEMENT Success (Documentation*)>
3744 <!ATTLIST Success
3745     fromBusinessState CDATA #REQUIRED
3746     fromBusinessStateIDRef IDREF #IMPLIED
3747     guardCondition (Success | BusinessFailure | TechnicalFailure |
3748 AnyFailure ) #IMPLIED
3749     guardExpression CDATA #IMPLIED
3750 >
3751 <!-- Failure is a special type of Transition in that it only has a
3752 origination -->
3753 <!-- guardExpression is an expression (e.g. XPath on BusinessDocument)
3754 that results in a boolean result -->
3755 <!ELEMENT Failure (Documentation*)>
3756 <!ATTLIST Failure
3757     fromBusinessState CDATA #REQUIRED
3758     fromBusinessStateIDRef IDREF #IMPLIED
3759     guardCondition (Success | BusinessFailure | TechnicalFailure |
3760 AnyFailure ) #IMPLIED
3761     guardExpression CDATA #IMPLIED
3762 >
3763
```

```
3764 <!-- Fork is a special type of BusinessState that can be transitioned
3765 to -->
3766 <!ELEMENT Fork (Documentation*)>
3767 <!ATTLIST Fork
3768     name CDATA #REQUIRED
3769     nameID ID #IMPLIED
3770 >
3771 <!-- Fork is a special type of BusinessState that can be transitioned
3772 to -->
3773 <!ELEMENT Join (Documentation*)>
3774 <!ATTLIST Join
3775     name CDATA #REQUIRED
3776     nameID ID #IMPLIED
3777     waitForAll (true | false) "true"
3778 >
3779
3780 <!-- fromAuthorizedRole and toAuthorizedRole are fully qualified using
3781 XPath -->
3782 <!-- BusinessTransactionActivity is a BusinessState that can be
3783 transitioned to -->
3784 <!ELEMENT BusinessTransactionActivity (Documentation*)>
3785 <!ATTLIST BusinessTransactionActivity
3786     name CDATA #REQUIRED
3787     nameID ID #IMPLIED
3788     businessTransaction CDATA #REQUIRED
3789     businessTransactionIDRef IDREF #IMPLIED
3790     fromAuthorizedRole CDATA #REQUIRED
3791     fromAuthorizedRoleIDRef IDREF #IMPLIED
3792     toAuthorizedRole CDATA #REQUIRED
3793     toAuthorizedRoleIDRef IDREF #IMPLIED
3794     isConcurrent (true | false) "true"
3795     isLegallyBinding (true | false) "true"
3796     timeToPerform CDATA #IMPLIED
3797 >
3798
3799 <!-- fromAuthorizedRole and toAuthorizedRole are fully qualified using
3800 XPath -->
3801 <!-- CollaborationActivity is a BusinessState that can be transitioned
3802 to -->
3803 <!ELEMENT CollaborationActivity (Documentation*)>
3804 <!ATTLIST CollaborationActivity
3805     name CDATA #REQUIRED
3806     nameID ID #IMPLIED
3807     fromAuthorizedRole CDATA #REQUIRED
3808     fromAuthorizedRoleIDRef IDREF #IMPLIED
3809     toAuthorizedRole CDATA #REQUIRED
3810     toAuthorizedRoleIDRef IDREF #IMPLIED
3811     binaryCollaboration CDATA #REQUIRED
3812     binaryCollaborationIDRef IDREF #IMPLIED
3813 >
3814 <!ELEMENT BusinessTransaction (Documentation*,
3815 RequestingBusinessActivity, RespondingBusinessActivity)>
3816 <!ATTLIST BusinessTransaction
3817     name CDATA #REQUIRED
3818     nameID ID #IMPLIED
```

```
3819     pattern CDATA #IMPLIED
3820     beginsWhen CDATA #IMPLIED
3821     endsWhen CDATA #IMPLIED
3822     isGuaranteedDeliveryRequired (true | false) "false"
3823     preCondition CDATA #IMPLIED
3824     postCondition CDATA #IMPLIED
3825 >
3826 <!--ELEMENT RequestingBusinessActivity (Documentation*,
3827 DocumentEnvelope)>
3828 <!--ATTLIST RequestingBusinessActivity
3829     name CDATA #IMPLIED
3830     nameID ID #IMPLIED
3831     isAuthorizationRequired (true | false) "false"
3832     isIntelligibleCheckRequired (true | false) "false"
3833     isNonRepudiationReceiptRequired (true | false) "false"
3834     isNonRepudiationRequired (true | false) "false"
3835     timeToAcknowledgeAcceptance CDATA #IMPLIED
3836     timeToAcknowledgeReceipt CDATA #IMPLIED
3837 >
3838 <!--ELEMENT RespondingBusinessActivity (Documentation*,
3839 DocumentEnvelope*)>
3840 <!--ATTLIST RespondingBusinessActivity
3841     name CDATA #IMPLIED
3842     nameID ID #IMPLIED
3843     isAuthorizationRequired (true | false) "false"
3844     isIntelligibleCheckRequired (true | false) "false"
3845     isNonRepudiationReceiptRequired (true | false) "false"
3846     isNonRepudiationRequired (true | false) "false"
3847     timeToAcknowledgeReceipt CDATA #IMPLIED
3848 >
3849 <!-- isPositiveResponse is an expression (e.g. XPath on
3850 BusinessDocument) that results in a boolean result -->
3851 <!--ELEMENT DocumentEnvelope (Documentation*, Attachment*)>
3852 <!--ATTLIST DocumentEnvelope
3853     businessDocument CDATA #REQUIRED
3854     businessDocumentIDRef IDREF #IMPLIED
3855     isPositiveResponse CDATA #IMPLIED
3856     isAuthenticated (true | false) "false"
3857     isConfidential (true | false) "false"
3858     isTamperProof (true | false) "false"
3859 >
3860 <!--ELEMENT Attachment (Documentation*)>
3861 <!--ATTLIST Attachment
3862     name CDATA #REQUIRED
3863     nameID ID #IMPLIED
3864     businessDocument CDATA #IMPLIED
3865     businessDocumentIDRef IDREF #IMPLIED
3866     mimeType CDATA #REQUIRED
3867     specification CDATA #IMPLIED
3868     version CDATA #IMPLIED
3869     isAuthenticated (true | false) "false"
3870     isConfidential (true | false) "false"
3871     isTamperProof (true | false) "false"
3872 >
3873 <!--ELEMENT BusinessPartnerRole (Documentation*, Performs*, Transition*)>
```

```
3874 <!ATTLIST BusinessPartnerRole
3875     name CDATA #REQUIRED
3876     nameID ID #IMPLIED
3877 >
3878
3879 <!-- authorizedRole is fully qualified using XPath -->
3880 <!ELEMENT Performs (Documentation*)>
3881 <!ATTLIST Performs
3882     authorizedRole CDATA #REQUIRED
3883     authorizedRoleIDRef IDREF #IMPLIED
3884 >
```

3885 **Appendix C: Business Process Specification Schema**

3886 **XML Schema**

```
3887 <?xml version="1.0" encoding="UTF-8"?>
3888 <!-- edited with XML Spy v3.5 NT (http://www.xmlspy.com) by Kurt
3889 Kanaskie (Lucent Technologies) -->
3890 <!--W3C Schema generated by XML Spy v3.5 NT (http://www.xmlspy.com)-->
3891 <!-- Kanaskie Updated 2001-04-27
3892     This is the version that works with XML Spy 3.5
3893     Use uriReference instead of anyURI
3894     Use timeDuration instead of duration
3895 -->
3896 <!-- Kanaskie Changed 2001-04-27
3897     See DTD for list of changes.
3898     Differences from DTD version:
3899     AuthorizedRole minOccurs=2 maxOccurs=2
3900     <xsd:attribute name="pattern" type="xsd:anyURI"/>
3901     <xsd:attribute name="uri" type="xsd:anyURI" use="required"/>
3902     <xsd:attribute name="location" type="xsd:anyURI"/>
3903     <xsd:attribute name="logicalModel" type="xsd:anyURI"/>
3904     <xsd:attribute name="specification" type="xsd:anyURI"/>
3905     <xsd:attribute name="timeToPerform" type="xsd:duration"/>
3906     <xsd:attribute name="timeToPerform" type="xsd:duration"/>
3907     <xsd:attribute name="timeToAcknowledgeAcceptance"
3908 type="xsd:duration"/>
3909     <xsd:attribute name="timeToAcknowledgeReceipt"
3910 type="xsd:duration"/>
3911     <xsd:attribute name="timeToAcknowledgeAcceptance"
3912 type="xsd:duration"/>
3913     <xsd:attribute name="timeToAcknowledgeReceipt"
3914 type="xsd:duration"/>
3915     <xsd:attribute name="isAuthenticated" type="xsd:boolean"
3916 value="false"/>
3917     <xsd:attribute name="isConfidential" type="xsd:boolean"
3918 value="false"/>
3919     <xsd:attribute name="isTamperProof" type="xsd:boolean"
3920 value="false"/>
3921     <xsd:attribute name="isGuaranteedDeliveryRequired"
3922 type="xsd:boolean" value="false"/>
3923     <xsd:attribute name="isConcurrent" type="xsd:boolean"
3924 value="true"/>
3925     <xsd:attribute name="isLegallyBinding" type="xsd:boolean"
3926 value="true"/>
3927     <xsd:attribute name="isAuthenticated" type="xsd:boolean"
3928 value="false"/>
3929     <xsd:attribute name="isConfidential" type="xsd:boolean"
3930 value="false"/>
3931     <xsd:attribute name="isTamperProof" type="xsd:boolean"
3932 value="false"/>
3933     <xsd:attribute name="waitForAll" type="xsd:boolean"
3934 value="true"/>
3935     <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean"
3936 value="false"/>
```



```
3937     <xsd:attribute name="isIntelligibleCheckRequired"
3938 type="xsd:boolean" value="false"/>
3939     <xsd:attribute name="isNonRepudiationReceiptRequired"
3940 type="xsd:boolean" value="false"/>
3941     <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean"
3942 value="false"/>
3943     <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean"
3944 value="false"/>
3945     <xsd:attribute name="isIntelligibleCheckRequired"
3946 type="xsd:boolean" value="false"/>
3947     <xsd:attribute name="isNonRepudiationReceiptRequired"
3948 type="xsd:boolean" value="false"/>
3949     <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean"
3950 value="false"/>
3951     <xsd:attribute name="onInitiation" type="xsd:boolean"
3952 value="false"/>
3953 -->
3954 <xsd:schema targetNamespace="http://www.ebxml.org/BusinessProcess"
3955 xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
3956 xmlns="http://www.ebxml.org/BusinessProcess"
3957 elementFormDefault="qualified">
3958     <xsd:element name="Attachment">
3959         <xsd:complexType>
3960             <xsd:sequence>
3961                 <xsd:element ref="Documentation" minOccurs="0"
3962 maxOccurs="unbounded"/>
3963             </xsd:sequence>
3964             <xsd:attribute name="name" type="xsd:string"
3965 use="required"/>
3966             <xsd:attribute name="nameID" type="xsd:ID"/>
3967             <xsd:attribute name="businessDocument"
3968 type="xsd:string"/>
3969             <xsd:attribute name="businessDocumentIDRef"
3970 type="xsd:IDREF"/>
3971             <xsd:attribute name="specification"
3972 type="xsd:uriReference"/>
3973             <xsd:attribute name="mimeType" type="xsd:string"
3974 use="required"/>
3975             <xsd:attribute name="version" type="xsd:string"/>
3976             <xsd:attribute name="isAuthenticated"
3977 type="xsd:boolean" value="false"/>
3978             <xsd:attribute name="isConfidential"
3979 type="xsd:boolean" value="false"/>
3980             <xsd:attribute name="isTamperProof"
3981 type="xsd:boolean" value="false"/>
3982         </xsd:complexType>
3983     </xsd:element>
3984     <xsd:element name="AuthorizedRole">
3985         <xsd:complexType>
3986             <xsd:sequence>
3987                 <xsd:element ref="Documentation" minOccurs="0"
3988 maxOccurs="unbounded"/>
3989             </xsd:sequence>
3990             <xsd:attribute name="name" type="xsd:string"
3991 use="required"/>
```

```

3992         <xsd:attribute name="nameID" type="xsd:ID"/>
3993     </xsd:complexType>
3994 </xsd:element>
3995 <xsd:element name="BinaryCollaboration">
3996     <xsd:complexType>
3997         <xsd:sequence>
3998             <xsd:element ref="Documentation" minOccurs="0"
3999 maxOccurs="unbounded"/>
4000             <xsd:element ref="AuthorizedRole"/>
4001             <xsd:element ref="AuthorizedRole"/>
4002             <xsd:choice minOccurs="0"
4003 maxOccurs="unbounded">
4004                 <xsd:element ref="Documentation"
4005 minOccurs="0" maxOccurs="unbounded"/>
4006                 <xsd:element ref="Start"/>
4007                 <xsd:element ref="Transition"/>
4008                 <xsd:element ref="Success"/>
4009                 <xsd:element ref="Failure"/>
4010                 <xsd:element
4011 ref="BusinessTransactionActivity"/>
4012                 <xsd:element
4013 ref="CollaborationActivity"/>
4014                 <xsd:element ref="Fork"/>
4015                 <xsd:element ref="Join"/>
4016             </xsd:choice>
4017         </xsd:sequence>
4018         <xsd:attribute name="name" type="xsd:string"
4019 use="required"/>
4020         <xsd:attribute name="nameID" type="xsd:ID"/>
4021         <xsd:attribute name="pattern"
4022 type="xsd:uriReference"/>
4023         <xsd:attribute name="beginsWhen" type="xsd:string"/>
4024         <xsd:attribute name="endsWhen" type="xsd:string"/>
4025         <xsd:attribute name="preCondition"
4026 type="xsd:string"/>
4027         <xsd:attribute name="postCondition"
4028 type="xsd:string"/>
4029         <xsd:attribute name="timeToPerform"
4030 type="xsd:timeDuration"/>
4031     </xsd:complexType>
4032 </xsd:element>
4033 <xsd:element name="BusinessDocument">
4034     <xsd:complexType>
4035         <xsd:sequence>
4036             <xsd:element ref="Documentation" minOccurs="0"
4037 maxOccurs="unbounded"/>
4038         </xsd:sequence>
4039         <xsd:attribute name="name" type="xsd:string"
4040 use="required"/>
4041         <xsd:attribute name="nameID" type="xsd:ID"/>
4042     </xsd:complexType>
4043 </xsd:element>
4044 <xsd:element name="BusinessPartnerRole">
4045     <xsd:complexType>
4046         <xsd:sequence>

```

```
4047         <xsd:element ref="Documentation" minOccurs="0"
4048 maxOccurs="unbounded" />
4049         <xsd:element ref="Performs" minOccurs="0"
4050 maxOccurs="unbounded" />
4051         <xsd:element ref="Transition" minOccurs="0"
4052 maxOccurs="unbounded" />
4053     </xsd:sequence>
4054     <xsd:attribute name="name" type="xsd:string"
4055 use="required" />
4056     <xsd:attribute name="nameID" type="xsd:ID" />
4057 </xsd:complexType>
4058 </xsd:element>
4059 <xsd:element name="BusinessTransaction">
4060     <xsd:complexType>
4061         <xsd:sequence>
4062             <xsd:element ref="Documentation" minOccurs="0"
4063 maxOccurs="unbounded" />
4064             <xsd:element ref="RequestingBusinessActivity" />
4065             <xsd:element ref="RespondingBusinessActivity" />
4066         </xsd:sequence>
4067         <xsd:attribute name="name" type="xsd:string"
4068 use="required" />
4069         <xsd:attribute name="nameID" type="xsd:ID" />
4070         <xsd:attribute name="pattern"
4071 type="xsd:uriReference" />
4072         <xsd:attribute name="beginsWhen" type="xsd:string" />
4073         <xsd:attribute name="endsWhen" type="xsd:string" />
4074         <xsd:attribute name="isGuaranteedDeliveryRequired"
4075 type="xsd:boolean" value="false" />
4076         <xsd:attribute name="preCondition"
4077 type="xsd:string" />
4078         <xsd:attribute name="postCondition"
4079 type="xsd:string" />
4080     </xsd:complexType>
4081 </xsd:element>
4082 <xsd:element name="BusinessTransactionActivity">
4083     <xsd:complexType>
4084         <xsd:sequence>
4085             <xsd:element ref="Documentation" minOccurs="0"
4086 maxOccurs="unbounded" />
4087         </xsd:sequence>
4088         <xsd:attribute name="name" type="xsd:string"
4089 use="required" />
4090         <xsd:attribute name="nameID" type="xsd:ID" />
4091         <xsd:attribute name="businessTransaction"
4092 type="xsd:string" use="required" />
4093         <xsd:attribute name="businessTransactionIDRef"
4094 type="xsd:IDREF" />
4095         <xsd:attribute name="fromAuthorizedRole"
4096 type="xsd:string" use="required" />
4097         <xsd:attribute name="fromAuthorizedRoleIDRef"
4098 type="xsd:IDREF" />
4099         <xsd:attribute name="toAuthorizedRole"
4100 type="xsd:string" use="required" />
```

```
4101         <xsd:attribute name="toAuthorizedRoleIDRef"
4102 type="xsd:IDREF" />
4103         <xsd:attribute name="isConcurrent" type="xsd:boolean"
4104 value="true" />
4105         <xsd:attribute name="isLegallyBinding"
4106 type="xsd:boolean" value="true" />
4107         <xsd:attribute name="timeToPerform"
4108 type="xsd:timeDuration" />
4109     </xsd:complexType>
4110 </xsd:element>
4111 <xsd:element name="CollaborationActivity">
4112     <xsd:complexType>
4113         <xsd:sequence>
4114             <xsd:element ref="Documentation" minOccurs="0"
4115 maxOccurs="unbounded" />
4116         </xsd:sequence>
4117         <xsd:attribute name="nameID" type="xsd:ID" />
4118         <xsd:attribute name="name" type="xsd:string"
4119 use="required" />
4120         <xsd:attribute name="fromAuthorizedRole"
4121 type="xsd:string" use="required" />
4122         <xsd:attribute name="fromAuthorizedRoleIDRef"
4123 type="xsd:IDREF" />
4124         <xsd:attribute name="toAuthorizedRole"
4125 type="xsd:string" use="required" />
4126         <xsd:attribute name="toAuthorizedRoleIDRef"
4127 type="xsd:IDREF" />
4128         <xsd:attribute name="binaryCollaboration"
4129 type="xsd:string" use="required" />
4130         <xsd:attribute name="binaryCollaborationIDRef"
4131 type="xsd:IDREF" />
4132     </xsd:complexType>
4133 </xsd:element>
4134 <xsd:element name="DocumentEnvelope">
4135     <xsd:complexType>
4136         <xsd:sequence>
4137             <xsd:element ref="Documentation" minOccurs="0"
4138 maxOccurs="unbounded" />
4139             <xsd:element ref="Attachment" minOccurs="0"
4140 maxOccurs="unbounded" />
4141         </xsd:sequence>
4142         <xsd:attribute name="businessDocument"
4143 type="xsd:string" use="required" />
4144         <xsd:attribute name="businessDocumentIDRef"
4145 type="xsd:IDREF" />
4146         <xsd:attribute name="isPositiveResponse"
4147 type="xsd:string" />
4148         <xsd:attribute name="isAuthenticated"
4149 type="xsd:boolean" value="false" />
4150         <xsd:attribute name="isConfidential"
4151 type="xsd:boolean" value="false" />
4152         <xsd:attribute name="isTamperProof"
4153 type="xsd:boolean" value="false" />
4154     </xsd:complexType>
4155 </xsd:element>
```

```
4156     <xsd:element name="DocumentSpecification">
4157         <xsd:complexType>
4158             <xsd:sequence>
4159                 <xsd:element ref="Documentation" minOccurs="0"
4160 maxOccurs="unbounded"/>
4161                 <xsd:element ref="BusinessDocument"
4162 minOccurs="0" maxOccurs="unbounded"/>
4163             </xsd:sequence>
4164             <xsd:attribute name="name" type="xsd:string"
4165 use="required"/>
4166             <xsd:attribute name="nameID" type="xsd:ID"/>
4167             <xsd:attribute name="location"
4168 type="xsd:uriReference"/>
4169             <xsd:attribute name="logicalModel"
4170 type="xsd:uriReference"/>
4171             <xsd:attribute name="version" type="xsd:string"/>
4172         </xsd:complexType>
4173     </xsd:element>
4174     <xsd:element name="Documentation">
4175         <xsd:complexType>
4176             <xsd:simpleContent>
4177                 <xsd:restriction base="xsd:string">
4178                     <xsd:attribute name="uri"
4179 type="xsd:uriReference"/>
4180                 </xsd:restriction>
4181             </xsd:simpleContent>
4182         </xsd:complexType>
4183     </xsd:element>
4184     <xsd:element name="Failure">
4185         <xsd:complexType>
4186             <xsd:sequence>
4187                 <xsd:element ref="Documentation" minOccurs="0"
4188 maxOccurs="unbounded"/>
4189             </xsd:sequence>
4190             <xsd:attribute name="fromBusinessState"
4191 type="xsd:string" use="required"/>
4192             <xsd:attribute name="fromBusinessStateIDRef"
4193 type="xsd:IDREF"/>
4194             <xsd:attribute name="guardCondition">
4195                 <xsd:simpleType>
4196                     <xsd:restriction base="xsd:NMTOKEN">
4197                         <xsd:enumeration value="Success"/>
4198                         <xsd:enumeration
4199 value="BusinessFailure"/>
4200                         <xsd:enumeration
4201 value="TechnicalFailure"/>
4202                         <xsd:enumeration
4203 value="AnyFailure"/>
4204                     </xsd:restriction>
4205                 </xsd:simpleType>
4206             </xsd:attribute>
4207             <xsd:attribute name="guardExpression"
4208 type="xsd:string"/>
4209         </xsd:complexType>
4210     </xsd:element>
```

```
4211     <xsd:element name="Fork">
4212         <xsd:complexType>
4213             <xsd:sequence>
4214                 <xsd:element ref="Documentation" minOccurs="0"
4215 maxOccurs="unbounded"/>
4216             </xsd:sequence>
4217             <xsd:attribute name="name" type="xsd:string"
4218 use="required"/>
4219             <xsd:attribute name="nameID" type="xsd:ID"/>
4220         </xsd:complexType>
4221     </xsd:element>
4222     <xsd:element name="Include">
4223         <xsd:complexType>
4224             <xsd:sequence>
4225                 <xsd:element ref="Documentation" minOccurs="0"
4226 maxOccurs="unbounded"/>
4227             </xsd:sequence>
4228             <xsd:attribute name="name" type="xsd:string"
4229 use="required"/>
4230             <xsd:attribute name="uuid" type="xsd:string"
4231 use="required"/>
4232             <xsd:attribute name="uri" type="xsd:uriReference"
4233 use="required"/>
4234             <xsd:attribute name="version" type="xsd:string"
4235 use="required"/>
4236         </xsd:complexType>
4237     </xsd:element>
4238     <xsd:element name="Join">
4239         <xsd:complexType>
4240             <xsd:sequence>
4241                 <xsd:element ref="Documentation" minOccurs="0"
4242 maxOccurs="unbounded"/>
4243             </xsd:sequence>
4244             <xsd:attribute name="name" type="xsd:string"
4245 use="required"/>
4246             <xsd:attribute name="nameID" type="xsd:ID"/>
4247             <xsd:attribute name="waitForAll" type="xsd:boolean"
4248 value="true"/>
4249         </xsd:complexType>
4250     </xsd:element>
4251     <xsd:element name="MultiPartyCollaboration">
4252         <xsd:complexType>
4253             <xsd:sequence>
4254                 <xsd:element ref="Documentation" minOccurs="0"
4255 maxOccurs="unbounded"/>
4256                 <xsd:element ref="BusinessPartnerRole"
4257 minOccurs="0" maxOccurs="unbounded"/>
4258             </xsd:sequence>
4259             <xsd:attribute name="name" type="xsd:string"
4260 use="required"/>
4261             <xsd:attribute name="nameID" type="xsd:ID"/>
4262         </xsd:complexType>
4263     </xsd:element>
4264     <xsd:element name="Package">
4265         <xsd:complexType>
```

```
4266         <xsd:sequence>
4267             <xsd:element ref="Documentation" minOccurs="0"
4268 maxOccurs="unbounded" />
4269             <xsd:choice minOccurs="0"
4270 maxOccurs="unbounded">
4271                 <xsd:element ref="Package" />
4272                 <xsd:element ref="BinaryCollaboration" />
4273                 <xsd:element ref="BusinessTransaction" />
4274                 <xsd:element
4275 ref="MultiPartyCollaboration" />
4276             </xsd:choice>
4277         </xsd:sequence>
4278         <xsd:attribute name="name" type="xsd:string"
4279 use="required" />
4280         <xsd:attribute name="nameID" type="xsd:ID" />
4281     </xsd:complexType>
4282 </xsd:element>
4283 <xsd:element name="Performs">
4284     <xsd:complexType>
4285         <xsd:sequence>
4286             <xsd:element ref="Documentation" minOccurs="0"
4287 maxOccurs="unbounded" />
4288         </xsd:sequence>
4289         <xsd:attribute name="authorizedRole"
4290 type="xsd:string" use="required" />
4291         <xsd:attribute name="authorizedRoleIDRef"
4292 type="xsd:IDREF" />
4293     </xsd:complexType>
4294 </xsd:element>
4295 <xsd:element name="ProcessSpecification">
4296     <xsd:complexType>
4297         <xsd:sequence>
4298             <xsd:element ref="Documentation" minOccurs="0"
4299 maxOccurs="unbounded" />
4300             <xsd:choice minOccurs="0"
4301 maxOccurs="unbounded">
4302                 <xsd:element ref="Include" minOccurs="0"
4303 maxOccurs="unbounded" />
4304                 <xsd:element ref="DocumentSpecification"
4305 minOccurs="0" maxOccurs="unbounded" />
4306                 <xsd:element ref="ProcessSpecification"
4307 minOccurs="0" maxOccurs="unbounded" />
4308                 <xsd:element ref="Package" />
4309                 <xsd:element ref="BinaryCollaboration" />
4310                 <xsd:element ref="BusinessTransaction" />
4311                 <xsd:element
4312 ref="MultiPartyCollaboration" />
4313             </xsd:choice>
4314         </xsd:sequence>
4315         <xsd:attribute name="name" type="xsd:ID"
4316 use="required" />
4317         <xsd:attribute name="uuid" type="xsd:string"
4318 use="required" />
4319         <xsd:attribute name="version" type="xsd:string"
4320 use="required" />
```

```
4321         </xsd:complexType>
4322     </xsd:element>
4323     <xsd:element name="RequestingBusinessActivity">
4324         <xsd:complexType>
4325             <xsd:sequence>
4326                 <xsd:element ref="Documentation" minOccurs="0"
4327 maxOccurs="unbounded"/>
4328                 <xsd:element ref="DocumentEnvelope"/>
4329             </xsd:sequence>
4330             <xsd:attribute name="name" type="xsd:string"
4331 use="required"/>
4332             <xsd:attribute name="nameID" type="xsd:ID"/>
4333             <xsd:attribute name="isAuthorizationRequired"
4334 type="xsd:boolean" value="false"/>
4335             <xsd:attribute name="isIntelligibleCheckRequired"
4336 type="xsd:boolean" value="false"/>
4337             <xsd:attribute name="isNonRepudiationReceiptRequired"
4338 type="xsd:boolean" value="false"/>
4339             <xsd:attribute name="isNonRepudiationRequired"
4340 type="xsd:boolean" value="false"/>
4341             <xsd:attribute name="timeToAcknowledgeAcceptance"
4342 type="xsd:timeDuration"/>
4343             <xsd:attribute name="timeToAcknowledgeReceipt"
4344 type="xsd:timeDuration"/>
4345         </xsd:complexType>
4346     </xsd:element>
4347     <xsd:element name="RespondingBusinessActivity">
4348         <xsd:complexType>
4349             <xsd:sequence>
4350                 <xsd:element ref="Documentation" minOccurs="0"
4351 maxOccurs="unbounded"/>
4352                 <xsd:element ref="DocumentEnvelope"
4353 minOccurs="0" maxOccurs="unbounded"/>
4354             </xsd:sequence>
4355             <xsd:attribute name="name" type="xsd:string"
4356 use="required"/>
4357             <xsd:attribute name="nameID" type="xsd:ID"/>
4358             <xsd:attribute name="isAuthorizationRequired"
4359 type="xsd:boolean" value="false"/>
4360             <xsd:attribute name="isIntelligibleCheckRequired"
4361 type="xsd:boolean" value="false"/>
4362             <xsd:attribute name="isNonRepudiationReceiptRequired"
4363 type="xsd:boolean" value="false"/>
4364             <xsd:attribute name="isNonRepudiationRequired"
4365 type="xsd:boolean" value="false"/>
4366             <xsd:attribute name="timeToAcknowledgeReceipt"
4367 type="xsd:timeDuration"/>
4368         </xsd:complexType>
4369     </xsd:element>
4370     <xsd:element name="Start">
4371         <xsd:complexType>
4372             <xsd:sequence>
4373                 <xsd:element ref="Documentation" minOccurs="0"
4374 maxOccurs="unbounded"/>
4375             </xsd:sequence>
```



```
4376         <xsd:attribute name="toBusinessState"
4377 type="xsd:string" use="required" />
4378         <xsd:attribute name="toBusinessStateIDRef"
4379 type="xsd:IDREF" />
4380     </xsd:complexType>
4381 </xsd:element>
4382 <xsd:element name="Success">
4383     <xsd:complexType>
4384         <xsd:sequence>
4385             <xsd:element ref="Documentation" minOccurs="0"
4386 maxOccurs="unbounded" />
4387         </xsd:sequence>
4388         <xsd:attribute name="fromBusinessState"
4389 type="xsd:string" use="required" />
4390         <xsd:attribute name="fromBusinessStateIDRef"
4391 type="xsd:IDREF" />
4392         <xsd:attribute name="guardCondition">
4393             <xsd:simpleType>
4394                 <xsd:restriction base="xsd:NMTOKEN">
4395                     <xsd:enumeration value="Success" />
4396                     <xsd:enumeration
4397 value="BusinessFailure" />
4398                     <xsd:enumeration
4399 value="TechnicalFailure" />
4400                     <xsd:enumeration
4401 value="AnyFailure" />
4402                 </xsd:restriction>
4403             </xsd:simpleType>
4404         </xsd:attribute>
4405         <xsd:attribute name="guardExpression"
4406 type="xsd:string" />
4407     </xsd:complexType>
4408 </xsd:element>
4409 <xsd:element name="Transition">
4410     <xsd:complexType>
4411         <xsd:sequence>
4412             <xsd:element ref="Documentation" minOccurs="0"
4413 maxOccurs="unbounded" />
4414         </xsd:sequence>
4415         <xsd:attribute name="onInitiation" type="xsd:boolean"
4416 value="false" />
4417         <xsd:attribute name="fromBusinessState"
4418 type="xsd:string" />
4419         <xsd:attribute name="fromBusinessStateIDRef"
4420 type="xsd:IDREF" />
4421         <xsd:attribute name="toBusinessState"
4422 type="xsd:string" />
4423         <xsd:attribute name="toBusinessStateIDRef"
4424 type="xsd:IDREF" />
4425         <xsd:attribute name="guardCondition">
4426             <xsd:simpleType>
4427                 <xsd:restriction base="xsd:NMTOKEN">
4428                     <xsd:enumeration value="Success" />
4429                     <xsd:enumeration
4430 value="BusinessFailure" />
```

```
4431                                     <xsd:enumeration
4432 value="TechnicalFailure"/>
4433                                     <xsd:enumeration
4434 value="AnyFailure"/>
4435                                     </xsd:restriction>
4436                                     </xsd:simpleType>
4437                                     </xsd:attribute>
4438                                     <xsd:attribute name="guardExpression"
4439 type="xsd:string"/>
4440                                     </xsd:complexType>
4441                                     </xsd:element>
4442 </xsd:schema>
4443
```

4444 **11 References**

4445

4446

1. UN/CEFACT Modelling Methodology (CEFACT/TMWG/N090R9.1)

4447

2. RosettaNet Implementation Framework: Core Specification, Version:

4448

Release 2.00.00, 3 January 2001

4449

4450 **12 Disclaimer**

4451

The views and specification expressed in this document are those of the authors

4452

and are not necessarily those of their employers. The authors and their

4453

employers specifically disclaim responsibility for any problems arising from

4454

correct or incorrect implementation or use of this design.

4455 **13 Contact Information**

4456

4457 Team Leader (Of the BP team):

4458 Paul Levine

4459 Telcordia Technologies, Inc.

4460 45 Knightsbridge Road

4461 Piscataway, N.J. 08854

4462 US

4463

4464 Phone: 732-699-3042

4465 EMail: plevine@telcordia.com

4466

4467 Sub Team Lead (Of the context/MetamodelGroup) :

4468

4469 Karsten Riemer

4470 Sun Microsystems

4471 1 Network Drive

4472 Burlington, MA 01803

4473 USA

4474

4475 Phone: 781-442-2679

4476 EMail: karsten.riemer@sun.com

4477

4478 Editor (of this document):

4479

4480 Karsten Riemer

4481 Sun Microsystems

4482 1 Network Drive

4483 Burlington, MA 01803

4484 USA

4485

4486 Phone: 781-442-2679

4487 EMail: karsten.riemer@sun.com

4488

4489 Copyright Statement

4490 Copyright © ebXML 2001. All Rights Reserved.

4491

4492 This document and translations of it may be copied and furnished to others, and
4493 derivative works that comment on or otherwise explain it or assist in its implementation
4494 may be prepared, copied, published and distributed, in whole or in part, without
4495 restriction of any kind, provided that the above copyright notice and this paragraph are
4496 included on all such copies and derivative works. However, this document itself may not
4497 be modified in any way, such as by removing the copyright notice or references to
4498 ebXML, UN/CEFACT, or OASIS, except as required to translate it into languages other
4499 than English.

4500

4501 The limited permissions granted above are perpetual and will not be revoked by ebXML
4502 or its successors or assigns. This document and the information contained herein is
4503 provided on an "AS IS" basis and ebXML DISCLAIMS ALL WARRANTIES,
4504 EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY
4505 THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY
4506 RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR
4507 FITNESS FOR A PARTICULAR PURPOSE

4508