



Creating A Single Global Electronic Market

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

ebXML Business Process Specification Schema Version 1.01

Business Process Project Team

11 May 2001

1 Status of this Document

17
18
19
20
21
22
23
24
25
26
27
28
29

This Technical Specification document has been approved by the ebXML Plenary. This material fulfills requirements of the ebXML Requirements document. The formatting for this document is based on the Internet Society's Standard RFC format.

This version:

www.ebxml.org/specs/ebBPSS.pdf

Latest version:

www.ebxml.org/specs/ebBPSS.pdf

29 **2 ebXML BP/CoreComponents metamodel participants**

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

32

33 Team Lead:

34 Paul Levine, Telcordia

35

36 Editors:

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

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

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

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

41 Jamie Clark, McLure-Moynihan, Inc.: (Legal aspects)

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

43 John Yunker, Edifecs: (Signal structures)

44 Karsten Riemer, Sun Microsystems: (Overall Document)

45

46 Participants:

47 Antoine Lonjon, Mega

48 J.J. Dubray, Excelon

49 Bob Haugen, Logistical Software

50 Bill McCarthy, Michigan State University

51 Paul Levine, Telcordia

52 Brian Hayes, CommerceOne

53 Nita Sharma, Netfish

54 David Welsh, Nordstrom

55 Christopher Ferris, Sun Microsystems

56 Antonio Carrasco, Data Access Technologies

57

58

59

59	3	Table of Contents	
60	1	Status of this Document	i
61	2	ebXML BP/CoreComponents metamodel participants	ii
62	3	Table of Contents	iii
63	4	Introduction.....	v
64	4.1	Summary of Contents of Document	1
65	4.2	Audience	1
66	4.3	Related Documents	1
67	4.4	Prerequisites.....	2
68	5	Design Objectives	2
69	5.1	Goals/Objectives/Requirements/Problem Description	2
70	5.2	Caveats and Assumptions	3
71	5.2.1	Relationship between <i>ebXML Business Process Specification Schema</i> and UMM	3
72	6	System Overview	5
73	6.1	Key Concepts of the ebXML Business Process Specification Schema	10
74	6.2	How to use the ebXML Business Process Specification Schema	14
75	6.3	How ebXML Business Process Specification Schema is used with other ebXML	
76		specifications	14
77	6.4	How to design collaborations and transactions, re-using at design time	16
78	6.4.1	Specify a Business Transaction and its Business Document Flow.....	16
79	6.4.2	Specify a Binary Collaboration.....	22
80	6.4.3	Specify a MultiParty Collaboration	25
81	6.4.4	Specify a Choreography.....	27
82	6.4.5	The whole model.....	31
83	6.5	Core Business Transaction Semantics	34
84	6.5.1	Interaction Predictability.....	34
85	6.5.2	Creating legally binding contracts	37
86	6.5.3	Non-Repudiation.....	38
87	6.5.4	Authorization security.....	39
88	6.5.5	Document security	40
89	6.5.6	Reliability.....	41
90	6.5.7	Parameters required for CPP/CPA.....	41
91	6.6	Run time Business Transaction semantics	41
92	6.6.1	Timeouts.....	42
93	6.6.2	Exceptions	44
94	6.7	Runtime Collaboration Semantics	47
95	6.8	Where the ebXML Business Process Specification Schema May Be Implemented	47
96	7	UML Element Specification	47
97	7.1	Business Collaborations	47
98	7.1.1	MultiPartyCollaboration	47
99	7.1.2	BusinessPartnerRole	48
100	7.1.3	Performs	48
101	7.1.4	AuthorizedRole	49
102	7.1.5	BinaryCollaboration.....	50

103	7.1.6	BusinessActivity	51
104	7.1.7	BusinessTransactionActivity	51
105	7.1.8	CollaborationActivity.....	52
106	7.2	Business Transactions	52
107	7.2.1	BusinessTransaction.....	53
108	7.2.2	Business Action.....	54
109	7.2.3	RequestingBusinessActivity	55
110	7.2.4	RespondingBusinessActivity	55
111	7.3	Document flow.....	56
112	7.3.1	Document Security.....	56
113	7.3.2	Document Envelope	56
114	7.3.3	BusinessDocument.....	58
115	7.3.4	Attachment	58
116	7.4	Choreography within Collaborations.....	59
117	7.4.1	BusinessState	59
118	7.4.2	Transition.....	59
119	7.4.3	Start	60
120	7.4.4	CompletionState.....	60
121	7.4.5	Success.....	61
122	7.4.6	Failure	61
123	7.4.7	Fork.....	62
124	7.4.8	Join.....	62
125	7.5	Definition and Scope.....	63
126	7.6	Collaboration and transaction well-formedness rules	63
127	8	ebXML Business Process Specification Schema – (DTD).....	65
128	8.1	Documentation for the DTD	65
129	8.2	XML to UML cross-reference	94
130	8.3	Scoped Name Reference	96
131	8.4	Substitution Sets.....	97
132	8.5	Sample XML document against above DTD	97
133	9	Business signal structures	97
134	9.1.1	ReceiptAcknowledgment DTD.....	98
135	9.1.2	AcceptanceAcknowledgement DTD.....	100
136	9.1.3	Exception Signal DTD.....	101
137	10	Production Rules.....	103
138		Appendix A: Sample XML Business Process Specification	105
139		Appendix B: Business Process Specification Schema DTD.....	110
140		Appendix C: Business Process Specification Schema XML Schema	117
141	11	References	129
142	12	Disclaimer	129
143	13	Contact Information.....	130
144			

145 **4 Introduction**

146 **Executive Summary**

147

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

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

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

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

163 **4.1 Summary of Contents of Document**

164 This document describes the ebXML Specification Schema

165 This document describes the Specification Schema, both in its UML form and in
166 its DTD form.

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

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

173

174 **4.2 Audience**

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

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

181 The audience is not business application developers.

182 **4.3 Related Documents**

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

186

- 187 • ebXML Technical Architecture Specification, version 1.04
- 188 • ebXML Core Components Dictionary, version 1.04
- 189 • ebXML Naming Convention for Core Components, version 1.04
- 190 • ebXML Collaboration-Protocol Profile and Agreement Specification V1.0
- 191 • ebXML Business Process and Business Information Analysis Overview,
192 version 0.7
- 193 • ebXML Business Process Analysis Worksheets & Guidelines, version
194 0.10
- 195 • ebXML E-Commerce Patterns, version 0.99
- 196 • ebXML Catalog of Common Business Processes, version 0.99
- 197 • ebXML Message Service Specification V0.99

- 198 • UN/CEFACT Modeling Methodology (UMM) as defined in the N090R9.1
199 specification

200 **4.4 Prerequisites**

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

- 203 • Business process modeling techniques and principles
204 • The UML syntax and semantics
205 • The Extensible Markup Language (XML)

206 **5 Design Objectives**

207 **5.1 Goals/Objectives/Requirements/Problem Description**

208 Business process models describe interoperable business processes that allow
209 business partners to collaborate. Business process models for e-business must
210 be turned into software components that collaborate on behalf of the business
211 partners.

212 The goal of the ebXML Specification Schema is to provide the bridge between e-
213 business process modeling and specification of e-business software
214 components.

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

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

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

223 The UML version of the *ebXML Business Process Specification Schema* is
224 merely a UML Class Diagram. It is not intended for the direct creation of ebXML
225 Business Process Specifications. Rather, it is a self-contained statement of all
226 the specification elements and relationships required to be able to create an
227 ebXML compliant Business Process Specification. Any methodologies and/or
228 metamodels used for the creation of ebXML compliant Business Process
229 Specifications must at minimum support these elements and relationships.

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

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

236 **5.2 Caveats and Assumptions**

237 This specification is designed to specify the run time aspects of a business
238 collaboration.

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

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

246 **5.2.1 Relationship between *ebXML Business Process Specification*** 247 ***Schema* and UMM**

248
249 The UN/CEFACT Modeling Methodology (UMM) is a methodology for business
250 process and information modeling.

251 This section describes the relationship between UMM and the ebXML *Business*
252 *Process Specification Schema*.

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

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

276 The *ebXML Business Process Specification Schema* provides an additional view
277 of the UMM metamodel. This subset is provided to support the direct
278 specification of the nominal set of elements necessary to configure a runtime
279 system in order to execute a set of ebXML business transactions. By drawing

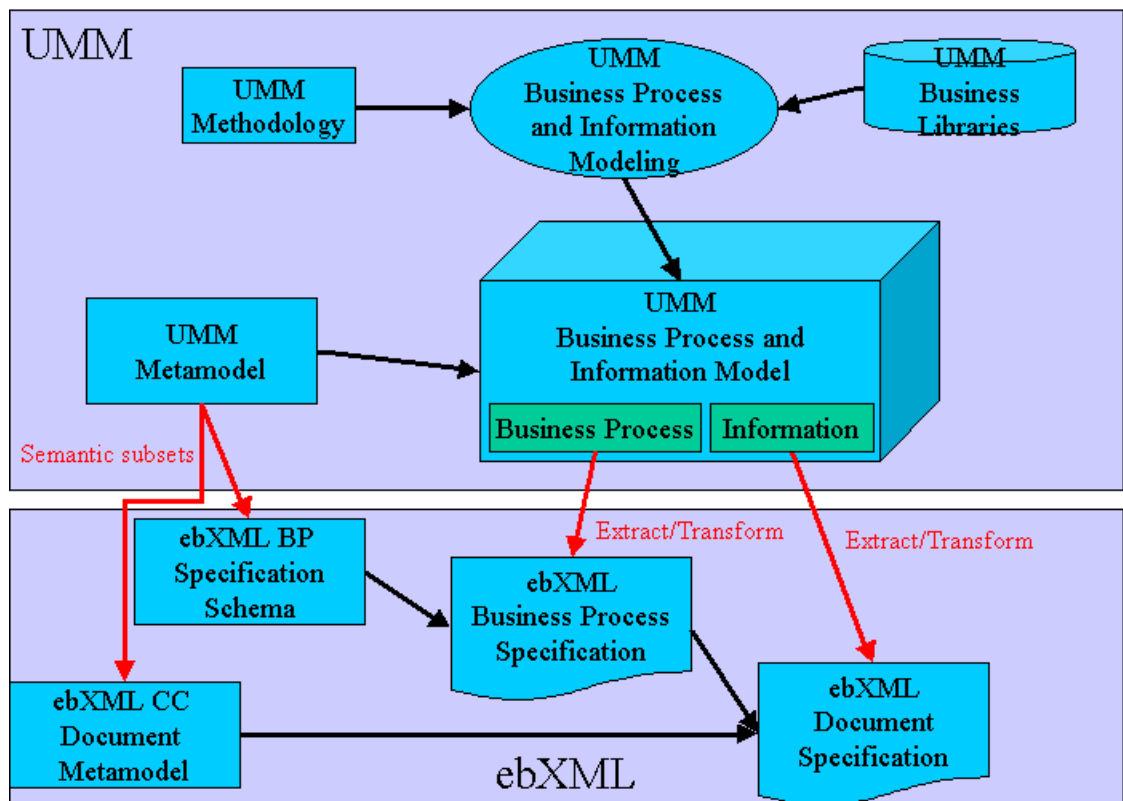
280 out modeling elements from several of the other views, the *ebXML Business*
 281 *Process Specification Schema* forms a semantic subset of the UMM Meta Model.
 282 Using the *ebXML Business Process Specification Schema* the user may thus
 283 create a Business Process Specification that contains only the information
 284 required to configure ebXML compliant software.

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

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

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

291



292

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

296 Since the *ebXML Business Process Specification Schema* is a semantic subset
 297 of the UMM metamodel, the user may then in an automated fashion extract from
 298 the Business Process and Information Model the required set of elements and

299 relationships, and transform them into an ebXML Business Process Specification
300 conforming to the *ebXML Business Process Specification Schema*.

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

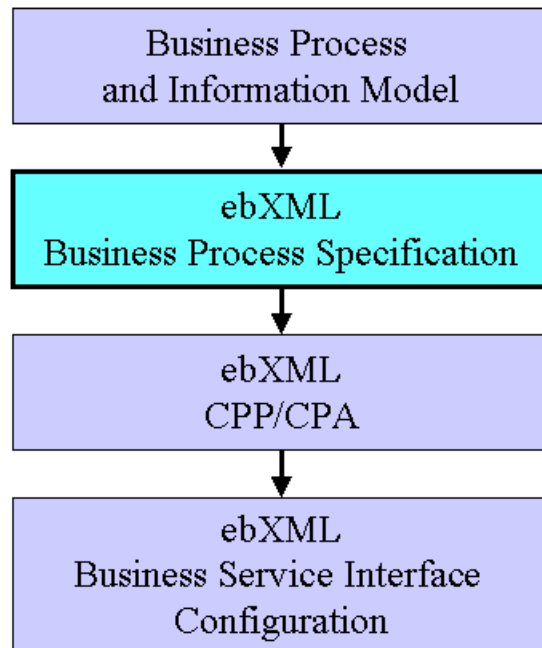
306 The UMM methodology is not part of the formal set of ebXML specifications.

307 Likewise, the UMM metamodel in its entirety is not part of the formal set of
308 ebXML specifications. Only the semantic subset represented by the *ebXML*
309 *Business Process Specification Schema* and CC are part of the formal set of
310 ebXML specifications.

311 The remainder of this document focuses on the *ebXML Business Process*
312 *Specification Schema* and Business Process Specifications created against it. It
313 is understood that proper Business Process and Information Modeling may have
314 taken place prior to beginning the activity of creating a Business Process
315 Specification.

316 **6 System Overview**

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



323

324 **Figure 2: Business Process Specification and Business Service Interface Configuration**

325 Using Business Process Modeling, a user may create a complete Business
326 Process and Information Model.

327 Based on this Business Process and Information Model and using the ebXML
328 *Business Process Specification Schema* the user will then extract and format the
329 nominal set of elements necessary to configure an ebXML runtime system in
330 order to execute a set of ebXML business transactions. The result is an ebXML
331 *Business Process Specification*.

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

334 An ebXML *Business Process Specification* contains the specification of Business
335 Transactions and the choreography of Business Transactions into Business
336 Collaborations.

337 This ebXML *Business Process Specification* is then the input to the formation of
338 ebXML trading partner Collaboration Protocol Profiles and Collaboration Protocol
339 Agreements.

340 These ebXML trading partner Collaboration Protocol Profiles and Collaboration
341 Protocol Agreements in turn serve as configuration files for ebXML Business
342 Service Interface software.

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

- 344
- 345 • UML version of the *Business Process Specification Schema*
 - 346 • XML version of the *Business Process Specification Schema*
 - 347 • Production Rules defining the mapping from the UML version of the *Business Process Specification Schema* to the XML version
 - 348
 - 349 • Business Signal Definitions

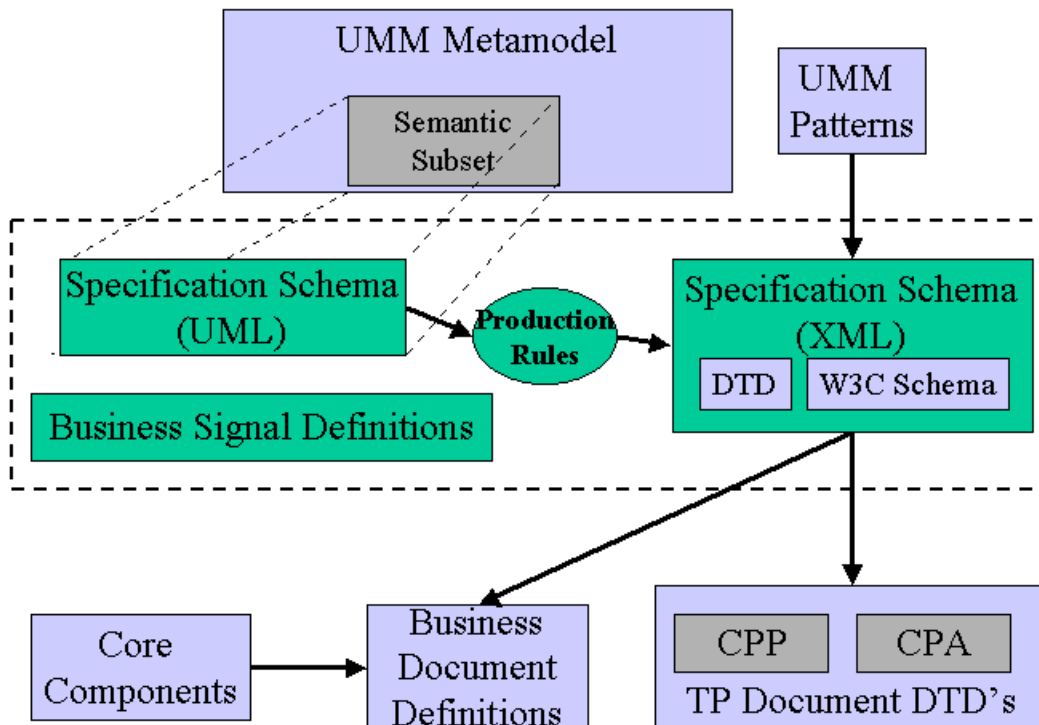
350

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

352

353 These components are shown (inside the dotted box) in figure 3 below.

354



355

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

358

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

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

363 The UML version of the *ebXML Business Process Specification Schema* is a
364 semantic subset of the metamodel behind UMM as specified in UN/CEFACT
365 TMWG's N090R9.1
366 N090R9.1 is as of this writing not yet approved by UN/CEFACT. It is the intent to
367 keep the *ebXML Business Process Specification Schema* and the UN/CEFACT
368 TMWG's N090 semantically aligned.

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

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

375 The XML version of the *ebXML Business Process Specification Schema* provides
376 the specification for XML based instances of *ebXML Business Process*
377 *Specifications*, and as a target for production rules from other representations.
378 Thus, a user may either create a *Business Process Specification* directly as an
379 XML document, or may chose to use some other means of specification first and
380 then apply production rules to arrive at the XML document version.

381 Any methodologies and/or metamodels used for the creation of ebXML compliant
382 Business Process Specifications must at minimum support the production of the
383 elements and relationships contained in the XML version of the *ebXML Business*
384 *Process Specification Schema*.

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

387 **UMM Business Process Interaction Patterns**

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

391 Each Business Transaction can be implemented using one of many available
392 standard patterns. These patterns determine the actual exchange of messages
393 and business signals between the partners to achieve the required electronic
394 commerce transaction.

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

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

403 **Business Signal Definitions**

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

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

410 The Business Process Specification Schema provides both the choreography of
411 business signals, and the structure definition of the business payload of a
412 business signal. The ebXML Message Service Specification signal structures
413 provide business service state alignment infrastructure, including unique
414 message identifiers and digests used to meet the basic process alignment
415 requirements. The business signal payload structures provided herein are
416 optional and normative and are intended to provide business and legal semantics
417 to the business signals.

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

419 **Production Rules**

420 A set of production rules are provided, defining the mapping from the UML
421 version of the ebXML *Business Process Specification Schema* to the XML
422 version.

423 The primary purpose for these production rules is to govern the one-time
424 generation of the DTD version of the ebXML *Business Process Specification
425 Schema* from the UML Class Diagram version of the ebXML *Business Process
426 Specification Schema*.

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

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

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

440 **Relationship to CPP/CPA**

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

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

449 ***Relationship to CC***

450 The *Business Process Specification Schema* does not by itself support the
451 definition of Business Documents. Rather, a *Business Process Specification*
452 merely points to the definition of Business Documents. Such definitions may
453 either be XML based, or – as attachments – may be any other structure, or
454 completely unstructured. XML based Business Document Specifications may be
455 based on the ebXML Core Components specifications.

456 ***Relationship to ebXML Message Service Specification***

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

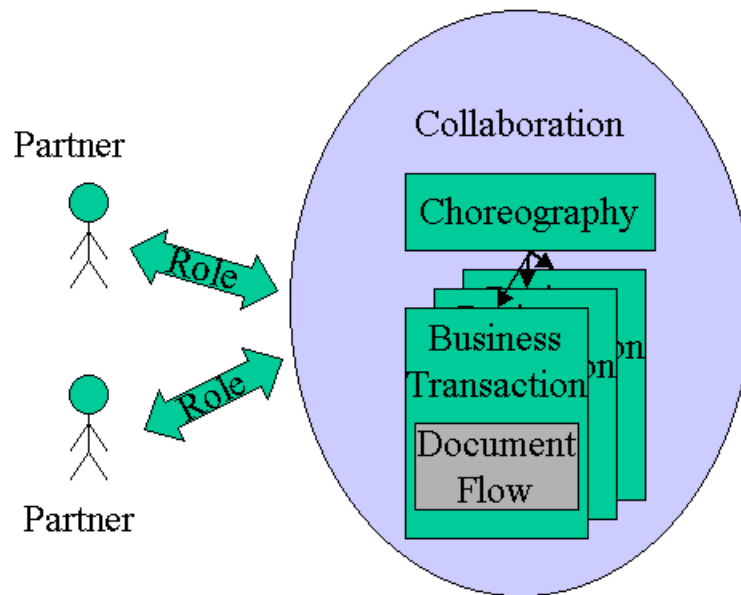
462

463 **6.1 Key Concepts of the ebXML Business Process Specification** 464 ***Schema***

465
466 The ebXML *Business Process Specification Schema* provides the semantics,
467 elements, and properties necessary to define business collaborations.

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

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



471

472

Figure 4. Basic Semantics of a business collaboration

473

474 Two or more business partners participate in the business collaboration through
 475 roles. The roles interact with each other through Business Transactions. The
 476 business transactions are sequenced relative to each other in a Choreography.
 477 Each Business Transaction consists of one or two predefined Business
 478 document flows. A Business Transaction may be additionally supported by one
 479 or more Business Signals.

480

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

482

1. Business Collaborations

483

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

486

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

489

Binary Collaborations are between two roles only.

490 Multiparty Collaborations are among more than two roles, but such
491 Multiparty Collaborations are always synthesized from two or more Binary
492 Collaborations. For instance if Roles A, B, and C collaborate and all
493 parties interact with each other, there will be a separate Binary
494 Collaboration between A and B, one between B and C, and one between
495 A and C. The Multiparty Collaboration will be the synthesis of these three
496 Binary Collaborations.

497 Binary Collaborations are expressed as a set of Business Activities
498 between the two roles. Each Business Activity reflects a state in the
499 collaboration. The Business Activity can be a Business Transaction
500 Activity, i.e. the activity of conducting a single Business Transaction, or a
501 Collaboration Activity, i.e. the activity of conducting another Binary
502 Collaboration. An example of the former is the activity of placing a
503 purchase order. An example of the latter is the activity of negotiating a
504 contract. In either case the activities can be choreographed relative to
505 other activities as per below.

506 The ability of a Binary Collaboration to have activities that in effect are
507 executing other Binary Collaborations, is the key to recursive
508 compositions of Binary Collaboration, and to the re-use of Binary
509 Collaborations.

510 In essence each Binary Collaboration is a re-useable protocol between
511 two roles.

512 2. Business Transactions

513 A Business Transaction is the atomic unit of work in a trading
514 arrangement between two business partners. A Business Transaction is
515 conducted between two parties playing opposite roles in the transaction.
516 The roles are always a requesting role and a responding role.

517 Like a Binary Collaboration, a Business Transaction is a re-useable
518 protocol between two roles. The way it is re-used is by referencing it from
519 a Binary Collaboration through the use of a Business Transaction Activity
520 as per above. In a Business Transaction Activity the roles of the Binary
521 Collaboration are assigned to the execution of the Business Transaction.

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

524 A Business Transaction is a very specialized and very constrained
525 protocol, in order to achieve very precise and enforceable transaction
526 semantics. These semantics are expected to be enforced by the software
527 managing the transaction, i.e. an ebXML Business Service Interface
528 (BSI).

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

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

574 **6.2 How to use the ebXML Business Process Specification** 575 **Schema**

576 The ebXML *Business Process Specification Schema* should be used wherever
577 ebXML compliant software is being specified to execute Business
578 Collaborations. The generic term for such software is a Business Service
579 Interface (BSI).

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

583 This section discusses

- 584 • How the ebXML *Business Process Specification Schema* fits in with other
585 ebXML specifications.
- 586 • How to use the ebXML *Business Process Specification Schema* at design
587 time, either for specifying brand new collaborations and transactions, or
588 for re-using existing ones.
- 589 • How to specify core transaction semantics and parameters needed for a
590 Collaboration-Protocol Profile and Agreement (CPP/CPA).
- 591 • Run-time transaction and collaboration semantics that the ebXML
592 *Business Process Specification Schema* specifies and the Business
593 Service Interface (BSI) is expected to manage.

594 **6.3 How ebXML Business Process Specification Schema is** 595 **used with other ebXML specifications**

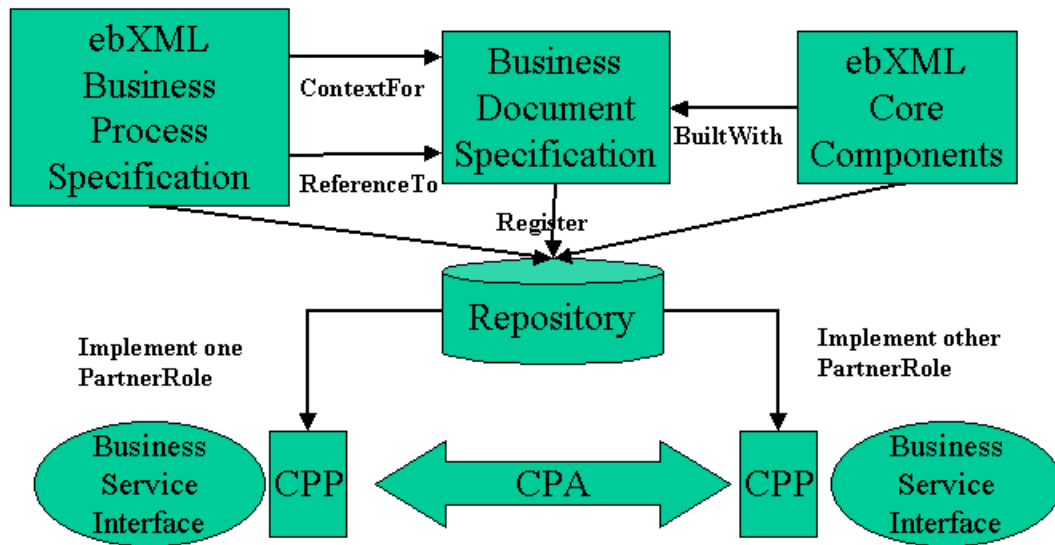
596 The ebXML *Business Process Specification Schema* provides the semantics,
597 elements, and properties necessary to define Business Collaborations.
598

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

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

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

613



614

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

616

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

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

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

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

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

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

636 **6.4 How to design collaborations and transactions, re-using at** 637 **design time**

638

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

- 642 1. Specify a Business Transaction
- 643 2. Specify the Business Document flow for a Business Transaction
- 644 3. Specify a Binary Collaboration re-using the Business Transaction
- 645 4. Specify a Choreography for the Binary Collaboration
- 646 5. Specify a higher level Binary Collaboration re-using the lower level Binary
647 Collaboration
- 648 6. Specify a Multiparty Collaboration re-using Binary Collaborations

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

653 The constructs listed above support the specification of fairly complex multi party
654 collaborations. However, an ebXML compliant Business Process Specificaton
655 may be as simple as a single Binary Collaboration referencing a single Business
656 Transaction. This involves only numbers 1 through 3 above. In other words,
657 Higher-level Binary Collaborations, Multi-party Collaborations and choreography
658 expressions are not required for ebXML Business Process Specification
659 compliance.

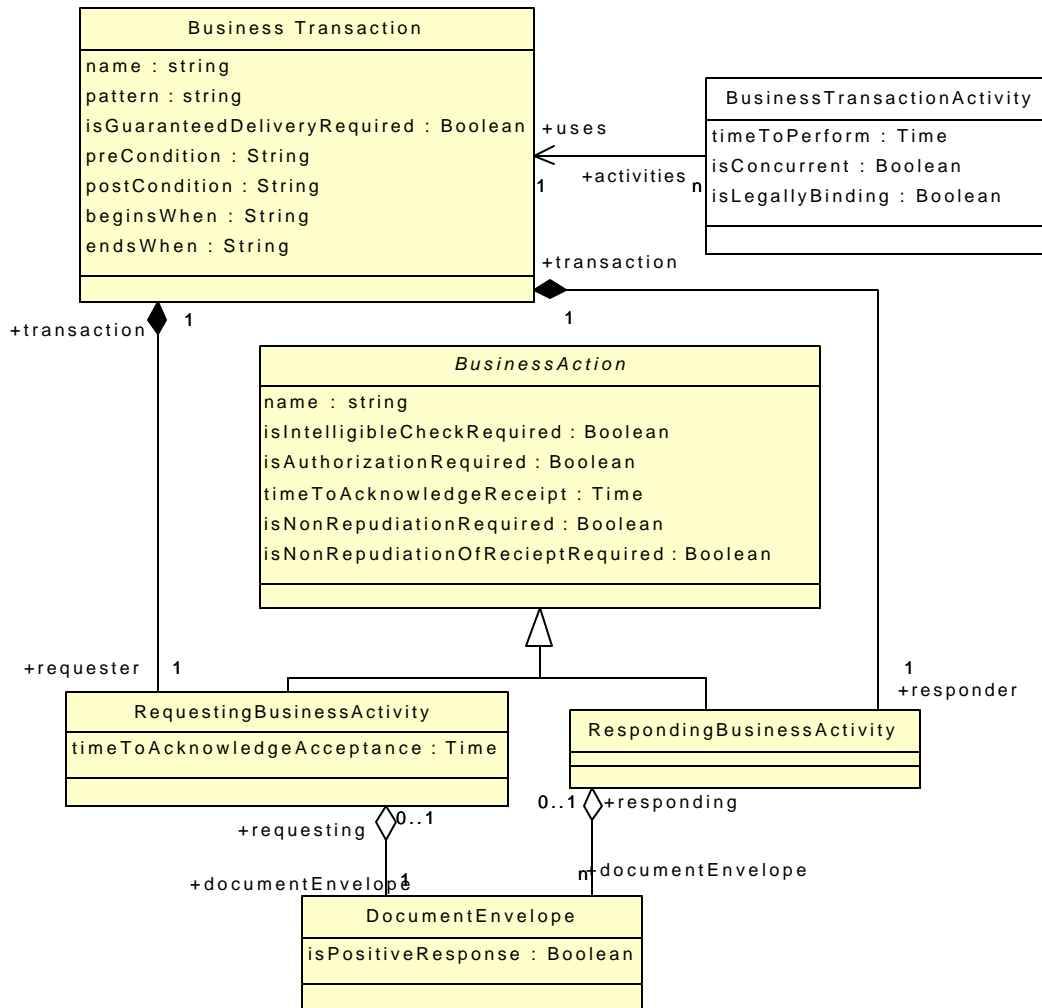
660

661

662 **6.4.1 Specify a Business Transaction and its Business Document** 663 **Flow**

664

665 Figure 6 illustrates a business transaction.



666

667

668

Figure 6. UML Diagram of a Business Transaction

669

670 6.4.1.1 Key Semantics of a Business Transaction

671

672

673

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

674

675

676

677

678

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.

679 Implicitly there is a requesting role performing the Requesting Business
680 Activity and a responding role performing the Responding Business
681 Activity. These roles become explicit when the transaction is used within
682 a Business Transaction Activity within a Binary Collaboration.

683 There is always a Request document flow.

684 Whether a Response document is required is part of the definition of the
685 Business Transaction. Some Business Transactions need this type of
686 request and response, typically for the formation of a contract or
687 agreement. Other Business Transactions are more like notifications, and
688 have only a Request document flow.

689 An abstract superclass, Business Action, is the holder of attributes that
690 are common to both Requesting Business Activity and Responding
691 Business Activity.

692 6.4.1.2 Sample syntax

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

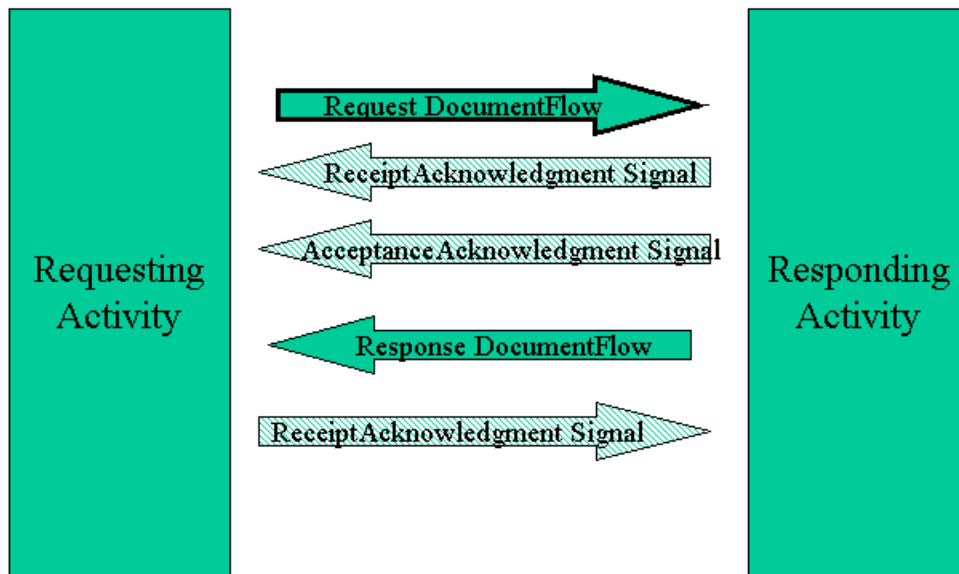
```
694 <BusinessTransaction name="Notify of advanceshipment">  
695     <RequestingBusinessActivity name="">  
696         <DocumentEnvelope  
697             BusinessDocument name="ASN"/>  
698     </RequestingBusinessActivity>  
699     <RespondingBusinessActivity name="">  
700     </RespondingBusinessActivity>  
701 </BusinessTransaction>
```

703 Associated with each document flow can be one or more business signals
704 acknowledging the document flow. These acknowledgment signals are
705 not modeled explicitly but parameters associated with the transaction
706 specify whether the signals are required or not.

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

709

710



711
712 Figure 7: Possible document flows and signals and their sequence

713

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

716

717

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

723

724

725

726

727

728

729

730

731

732

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.

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

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

742

743 6.4.1.3 Sample syntax

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

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

```
751 <BusinessTransaction name="Create Order">
752     <RequestingBusinessActivity name=" "
753         isNonRepudiationRequired="true"
754         timeToAcknowledgeReceipt="P2D"
755         timeToAcknowledgeAcceptance="P3D">
756         <DocumentEnvelope
757             BusinessDocument="Purchase Order"/>
758     </RequestingBusinessActivity>
759     <RespondingBusinessActivity name=" "
760         isNonRepudiationRequired="true"
761         timeToAcknowledgeReceipt="P5D">
762         <DocumentEnvelope isPositiveResponse="true"
763             BusinessDocument="PO
764             Acknowledgement"/>
765     </RespondingBusinessActivity>
766 </BusinessTransaction>
```

768

769

770 6.4.1.4 Specifying Business Document flows

771

772 Request document flows and response document flows contain Business
 773 Documents that pertain to the Business Transaction. The model for this is
 774 shown in Figure 8. Business Documents have varying structures.

775 Business signals, however always have the same structure, defined once
 776 and for all as part of the ebXML *Business Process Specification Schema*.

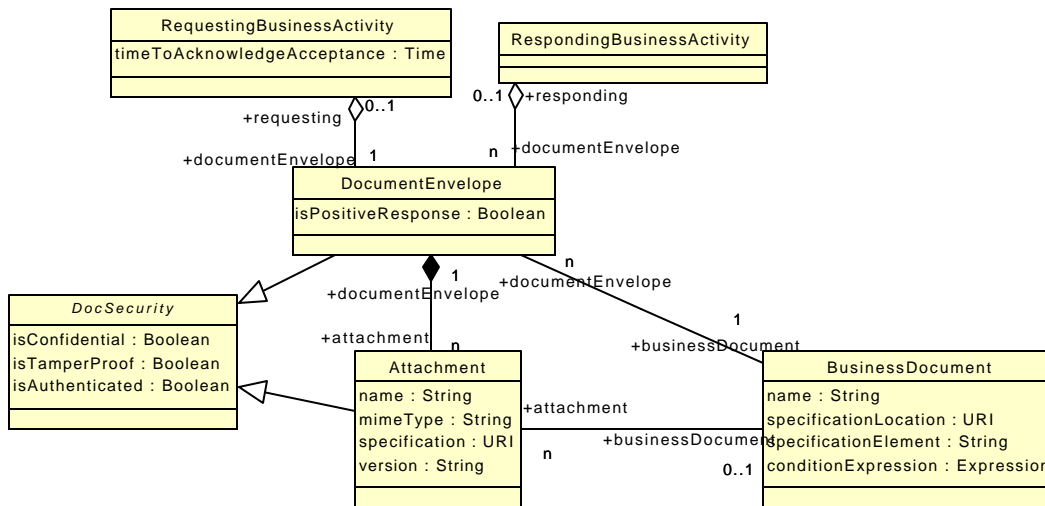
777

778

779

780

781



782

783

Figure 8: UML Diagram of document flow

784

785 A document flow is not modeled directly. Rather it is modeled indirectly as
 786 a Document Envelope sent by one role and received by the other. The
 787 Document Envelope is always associated with one Requesting Business
 788 Activity and one Responding Business Activity to model the flow.

789 Document Envelopes are named. There is always only one named
 790 Document Envelope for a Requesting Activity. There may be zero, one, or
 791 many mutually exclusive, named Document Envelopes for a Responding
 792 Activity. For example, the Response Document Envelopes for a purchase
 793 order transaction might be named PurchaseOrderAcceptance,
 794 PurchaseOrderDenial, and PartialPurchaseOrderAcceptance. In the
 795 actual execution of the purchase order transaction, however, only one of
 796 the defined possible responses will be sent.

797 The Document Envelope represents the flow of documents between the
 798 activities. Each Document Envelope carries exactly one primary Business
 799 Document.

800 A Document Envelope can optionally have one or more attachments, all
 801 related to the primary Business Document. The document and its
 802 attachments in essence form one transaction in the payload in the ebXML
 803 Message Service message structure.

804 6.4.1.5 Sample syntax

805 This example shows a business transaction with one request and two
 806 possible responses, a success and a failure. The request has an
 807 attachment. All the the Business Documents are fully qualified with the
 808 schema name.

809

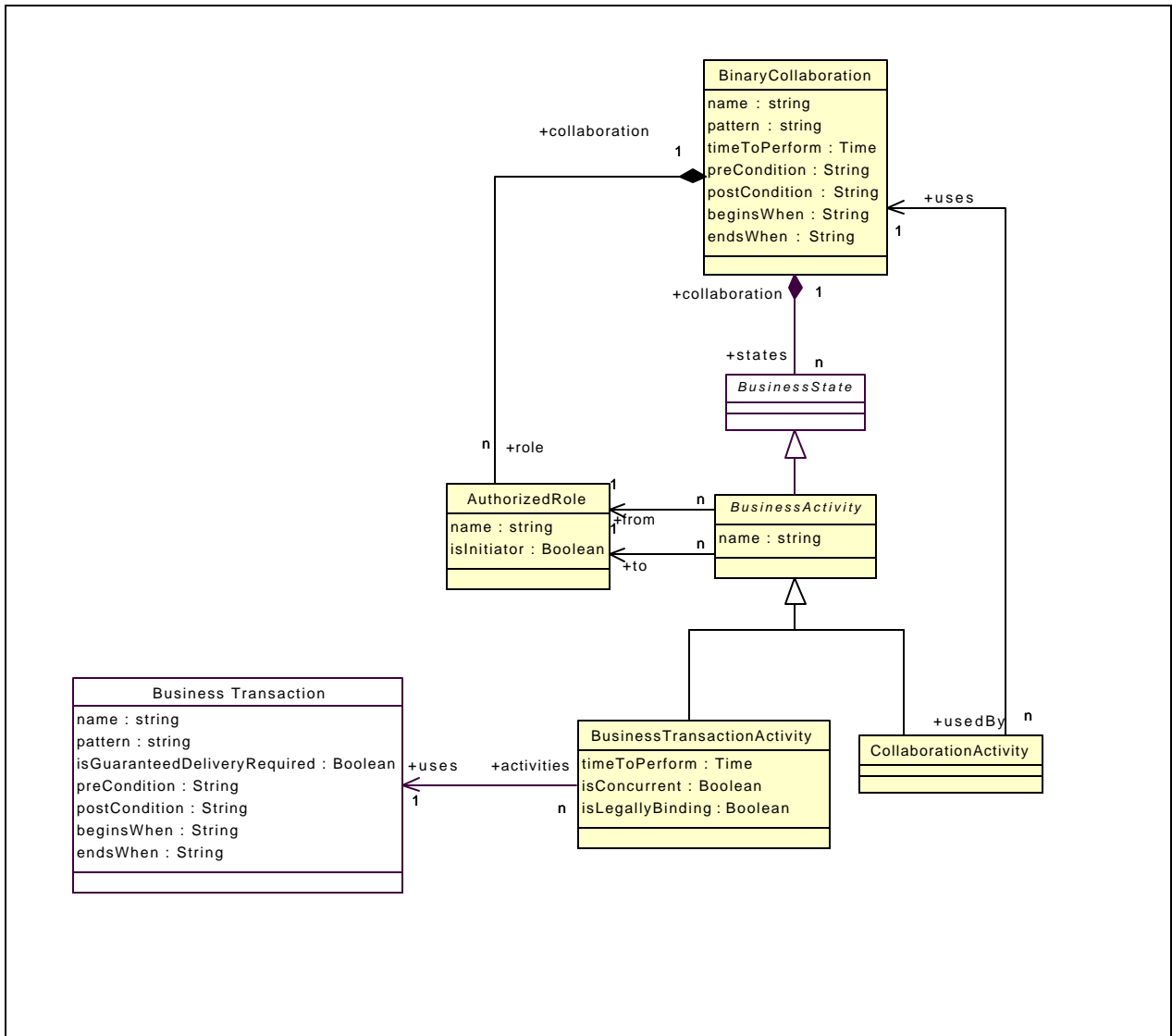
```

810 <BusinessDocument name=" Purchase Order "
811 specificationLocation="someplace"/>
812 <BusinessDocument name=" PO Acknowledgement "
813 specificationLocation="someplace"/>
814 <BusinessDocument name=" PO Rejection "
815 specificationLocation="someplace"/>
816 <BusinessDocument name="Delivery Instructions"
817 specificationLocation="someplace"/>
818
819 <BusinessTransaction name="Create Order">
820   <RequestingBusinessActivity name=""
821     <DocumentEnvelope isPositiveResponse="true"
822       BusinessDocument="ebXML1.0/PO Acknowledgement">
823       <Attachment
824         name="DeliveryNotes"
825         mimeType="XML"
826         BusinessDocument=
827         "ebXML1.0/Delivery Instructions"
828         specification=""
829         isConfidential="true"
830         isTamperProof="true"
831         isAuthenticated="true">
832       </Attachment>
833     </DocumentEnvelope>
834   </RequestingBusinessActivity>
835   <RespondingBusinessActivity name=""
836     <DocumentEnvelope
837       BusinessDocument="ebXML1.0/PO
838 Acknowledgement"/>
839     </DocumentEnvelope>
840     <DocumentEnvelope isPositiveResponse="false"
841       BusinessDocument=" ebXML1.0/PO Rejection"/>
842     </DocumentEnvelope>
843   </RespondingBusinessActivity>
844 </BusinessTransaction>
845

```

846 6.4.2 Specify a Binary Collaboration

847 Figure 9 illustrates a binary collaboration.



848

849

850

Figure 9: UML Diagram of a Binary Collaboration

851

852

853 6.4.2.1 Key Semantics of a Binary Collaboration

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

857 A Binary Collaboration consists of one or more Business Activities. These
858 Business Activities are always conducted **between** the two Authorized
859 Roles of the Binary Collaboration. For each activity one of two roles is
860 assigned to be the InitiatingRole (from) and the other to be the
861 RespondingRole (to).

862 A Business Activity can be either a Business Transaction Activity or a
863 Collaboration Activity.

864 A Business Transaction Activity is the performance of a Business
865 Transaction. Business Transactions are re-useable relative to Business
866 Transaction Activity. The same Business Transaction can be performed
867 by multiple Business Transaction Activities in different Binary
868 Collaborations, or even by multiple Business Transaction Activities in the
869 same Binary Collaboration.

870 A Collaboration Activity is the performance of a Binary Collaboration,
871 possibly within another Binary Collaboration. Binary Collaborations are re-
872 useable relative to Collaboration Activity. The same Binary Collaboration
873 can be performed by multiple Collaboration Activities in different Binary
874 Collaborations, or even by multiple Collaboration Activities in the same
875 Binary Collaboration.

876 When performing a Binary Collaboration within a Binary Collaboration
877 there is an implicit relationship between the roles at the two levels.
878 Assume that Binary Collaboration X is performing Binary Collaboration Y
879 through Collaboration Activity Q. Binary Collaboration X has Authorized
880 roles Customer and Vendor. In Collaboration Activity Q we assign
881 Customer to be the initiator, and Vendor to be the responder. Binary
882 Collaboration X has Authorized roles Buyer and Seller and a Business
883 Transaction Activity where Buyer is the initiator and Seller the responder.
884 We have now established a role relationship between the roles Customer
885 and Buyer because they are both initiators in activities in the related
886 performing and performed Binary Collaborations.

887 Since a Business Transaction is atomic in nature, the performing of a
888 single Business Transaction through a Business Transaction Activity is
889 also atomic in nature. If the desired semantic is not atomic, then the task
890 should be split over multiple transactions. For instance if it is desired to
891 model several partial acceptances of a request, then the request should
892 be modeled as one transaction within a binary collaboration and the
893 partial acceptance(s) as separate transactions.

894 The CPA/CPP Specification requires that parties agree upon a
895 Collaboration Protocol Agreement (CPA) in order to transact business. A
896 CPA associates itself with a specific Binary Collaboration. Thus, all
897 Business Transactions performed between two parties should be
898 referenced through Business Transaction Activities contained within a
899 Binary Collaboration.

900

901 6.4.2.2 Sample syntax

902

903 Here is a simple Binary Collaboration using one of the Business
904 Transactions defined above:

905

```

906     <BinaryCollaboration name="Firm Order"
907     timeToPerform="P2D">
908         <Documentation>
909             timeToPerform =
910             Period: 2 days from start of transaction
911         </Documentation>
912         <InitiatingRole name="buyer"/>
913         <RespondingRole name="seller"/>
914         <BusinessTransactionActivity name="Create Order"
915             businessTransaction="Create Order"
916             fromAuthorizedRole="buyer"
917             toAuthorizedRole="seller"/>
918     </BinaryCollaboration>

```

919

920 Here is a slightly more complex Binary Collaboration re-using the same
 921 Business Transaction as the previous Binary Collaboration, and adding the
 922 use of another of the Business Transactions defined above:

923

```

924     <BinaryCollaboration name="Product Fulfillment"
925     timeToPerform="P5D">
926         <Documentation>
927             timeToPerform =
928             Period: 5 days from start of transaction
929         </Documentation>
930         <InitiatingRole name="buyer"/>
931         <RespondingRole name="seller"/>
932         <BusinessTransactionActivity name="Create Order"
933             businessTransaction="Create Order"
934             fromAuthorizedRole="buyer"
935             toAuthorizedRole="seller"
936             isLegallyBinding="true" />
937         <BusinessTransactionActivity
938             name="Notify shipment"
939             businessTransaction="Notify of advance
940             shipment"
941             fromAuthorizedRole="buyer"
942             toAuthorizedRole="seller"/>
943     </BinaryCollaboration>

```

944

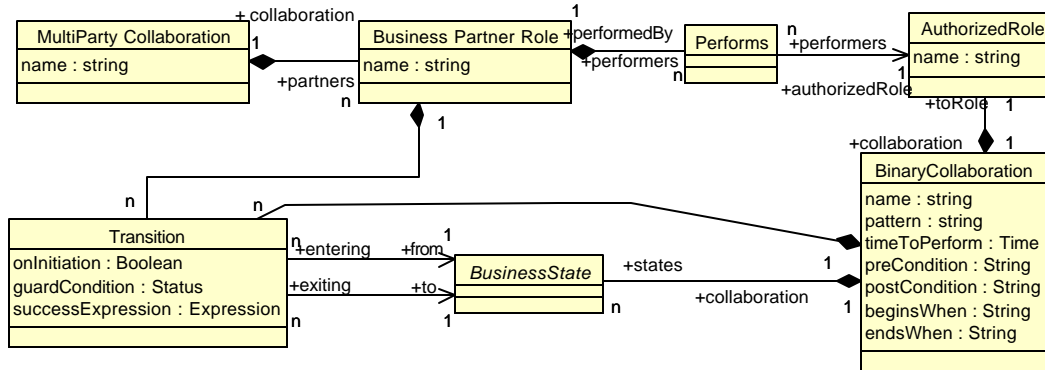
945

946

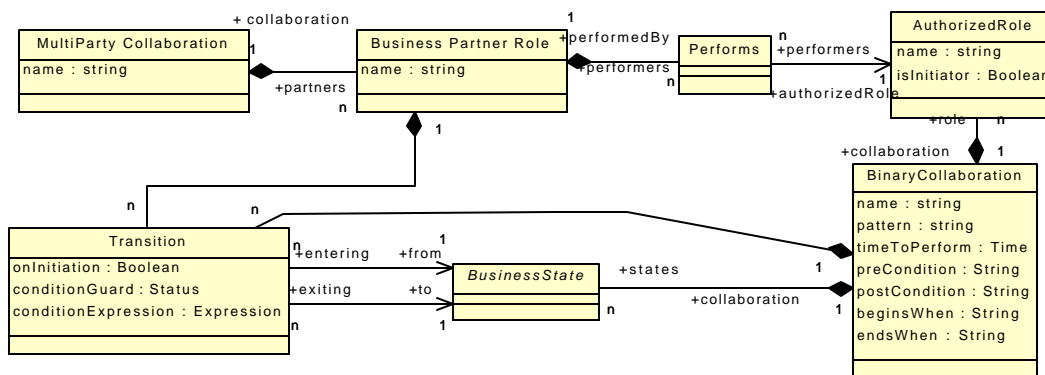
947 6.4.3 Specify a MultiParty Collaboration

948 Figure 10 illustrates a multiparty collaboration

949



950



951

952

Figure 10: UML Diagram of a MultiParty Collaboration

953

954

955 6.4.3.1 Key Semantics of a MultiParty Collaboration

956 A MultiParty Collaboration is a synthesis of Binary Collaborations.

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

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

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

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

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

972

973 6.4.3.2 Sample syntax

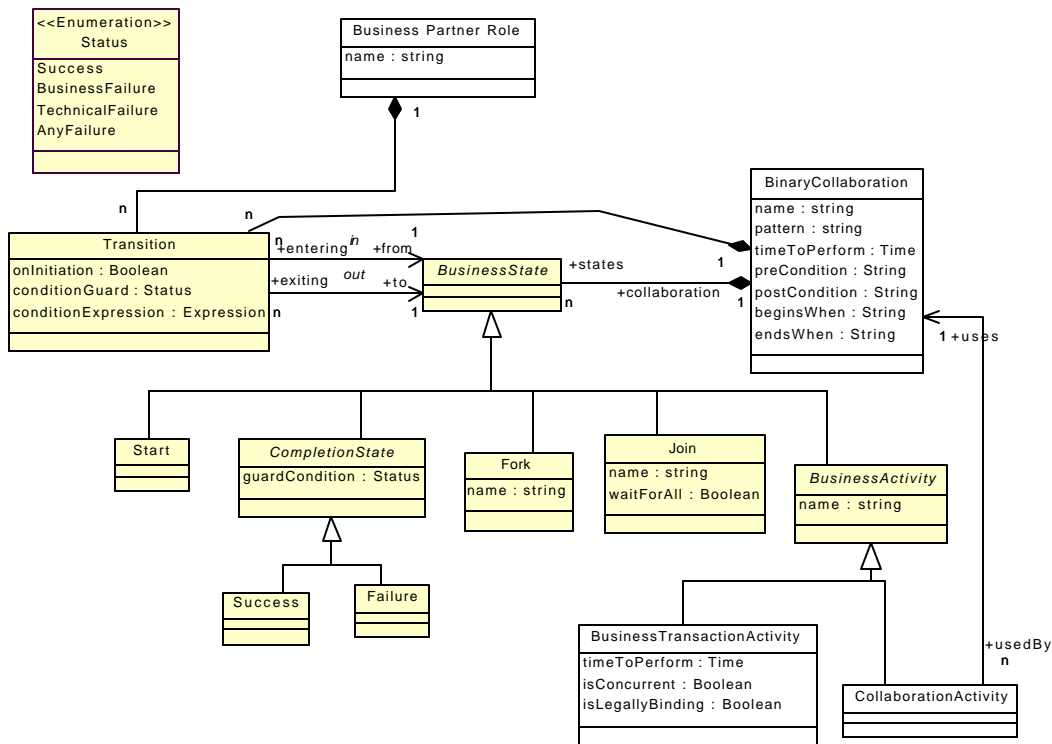
974 Here is a simple Multiparty Collaboration using the Binary Collaborations
975 defined above.

```
976           <MultiPartyCollaboration name="DropShip">
977           <BusinessPartnerRole name="Customer">
978            <Performs
979             initiatingRole=
980             '//binaryCollaboration[@name="Firm Order"]
981             /InitiatingRole[@name="buyer"]' />
982           </BusinessPartnerRole>
983           <BusinessPartnerRole name="Retailer">
984            <Performs
985             respondingRole=
986             '//binaryCollaboration[@name="Firm Order"]
987             /RespondingRole[@name="seller"]' />
988            <Performs
989             initiatingRole=
990             '//binaryCollaboration[@name=" Product
991             Fulfillment" /InitiatingRole[@name="buyer"]' />
992           </BusinessPartnerRole>
993           <BusinessPartnerRole name="DropShip Vendor">
994            <Performs
995             respondingRole=
996             '//binaryCollaboration[@name=" Product
997             Fulfillment"
998             /RespondingRole[@name="seller"]' />
999           </BusinessPartnerRole>
1000          </MultiPartyCollaboration>
```

1001

1002 6.4.4 Specify a Choreography

1003 Figure 11 illustrates a choreography.



1004

1005

1006

Figure 11: UML Diagram of a Choreography

1007

1008 6.4.4.1 Key Semantics of a Choreography

1009

1010 A Choreography is an ordering and sequencing of Business Activities
 1011 within a Binary Collaboration.

1012 The choreography is specified in terms of Business States, and
 1013 transitions between those Business States.

1014 A Business Activity is an abstract kind of Business State. Its two subtypes
 1015 Business Transaction Activity and Collaboration Activity are concrete
 1016 Business States. The purpose of a Choreography is to order and
 1017 sequence Business Transaction Activity and/or Collaboration Activity
 1018 within a Binary Collaboration, or across Binary Collaborations within a
 1019 Multiparty Collaboration.

1020 There are a number of auxiliary kinds of Business States that facilitate the
 1021 choreographing of Business Activities. These include a Start state, a
 1022 Completion state (which comes in a Success and Failure flavor), a Fork
 1023 state and a Synchronization state. These are all equivalent to
 1024 diagramming artifacts on a UML activity chart.

1025 Transitions are between Business States. Transitions can be gated by
 1026 Guards. Guards can refer to the status of the Document Envelope that
 1027 caused the transition, the type of Document sent, the content of the
 1028 document, or postconditions on the prior state.

1029 A Transition can also be used to create nested
 1030 BusinessTransactionActivities. A nested BusinessTransactionActivity is
 1031 one where a first transition happens after the receipt of the request in the
 1032 first transaction, and then the entire second transaction is performed
 1033 before returning to the first transaction to send the response back to the
 1034 original requestor. The flag 'onInitiation' in Transition is used for this
 1035 purpose. Nested BusinessTransactionActivity are typically within a
 1036 multiparty collaboration. In essence an Authorized Role in one Binary
 1037 Collaboration receives a request, then turns around and becomes the
 1038 requestor in an other Binary Collaboration before coming back and
 1039 sending the response in the first Binary Collaboration.

1040 isConcurrent is a parameter that governs the flow of transactions. Unlike
 1041 the security and timing parameters it does not govern the internal flow of
 1042 a transaction, rather it determines whether multiple instances of that
 1043 transaction type can be 'open' at the same time as part of the same
 1044 business transaction activity. IsConcurrent is the parameter that governs
 1045 this. It is at the business transaction activity level.

1046

1047 6.4.4.2 Sample syntax

1048
 1049 Here is the same Binary Collaboration as used before, with choreography
 1050 added at the end. There is a transition between the two, a start and two
 1051 possible outcomes of this collaboration, success and failure:

```

1052 <BinaryCollaboration name="Product Fulfillment"
1053   timeToPerform="P5D">
1054     <Documentation>
1055       timeToPerform =
1056         Period: 5 days from start of transaction
1057     </Documentation>
1058     <InitiatingRole name="buyer"/>
1059     <RespondingRole name="seller"/>
1060     <BusinessTransactionActivity name="Create Order"
1061       businessTransaction="Create Order"
1062       fromAuthorizedRole="buyer"
1063       toAuthorizedRole="seller"/>
1064     <BusinessTransactionActivity
1065       name="Notify shipment"
1066       businessTransaction="Notify of advance
1067       shipment"
1068       fromAuthorizedRole="buyer"
1069       toAuthorizedRole="seller"/>
1070     <Start toBusinessState="Create Order"/>
1071     <Transition
  
```

```

1072         fromBusinessState="Create Order"
1073         toBusinessState="Notify shipment"/>
1074     <Success fromBusinessState="Notify shipment"
1075         conditionGuard="Success"/>
1076     <Failure fromBusinessState="Notify shipment"
1077         conditionGuard="BusinessFailure"/>
1078 </BinaryCollaboration>
1079

```

1080 Here is the same Multipart Collaboration as defined before, but with a simple
1081 choreography (transition) across two Binary Collaborations.

```

1082
1083 <MultiPartyCollaboration name="DropShip">
1084     <BusinessPartnerRole name="Customer">
1085         <Performs
1086             initiatingRole=
1087                 \'//binaryCollaboration[@name="Firm Order"]
1088                 /InitiatingRole[@name="buyer"]\' />
1089     </BusinessPartnerRole>
1090     <BusinessPartnerRole name="Retailer">
1091         <Performs
1092             respondingRole=
1093                 \'//binaryCollaboration[@name="Firm Order"]
1094                 /RespondingRole[@name="seller"]\' />
1095         <Performs
1096             initiatingRole=
1097                 \'//binaryCollaboration[@name=" Product
1098                 Fulfillment" /initiatingRole[@name="buyer"]\' />
1099         <Transition
1100             fromBinaryCollaboration"Firm Order"
1101             fromBusinessState=
1102                 \'//binaryCollaboration[@name="Firm Order"]
1103                 /[@name="Create Order"]\'
1104             toBusinessState=
1105                 \'//binaryCollaboration[@name="Product
1106                 Fulfillment" ]
1107                 /[@name="Create Order"]\'
1108             />
1109     </BusinessPartnerRole>
1110     <BusinessPartnerRole name="DropShip Vendor">
1111         <Performs
1112             respondingRole=
1113                 \'//binaryCollaboration[@name=" Product
1114                 Fulfillment"
1115                 /RespondingRole[@name="seller"]\' />
1116     </BusinessPartnerRole>
1117 </MultiPartyCollaboration>
1118
1119

```

1120 6.4.5 The whole model

1121

1122

1123

1124

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*

1125
1126

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

1128

1129 **6.5 Core Business Transaction Semantics**

1130 The ebXML concept of a business transaction and the semantics behind it are
1131 central to predictable, enforceable commerce. It is expected that any Business
1132 Service Interface (BSI) will be capable of managing a transaction according to
1133 these semantics.

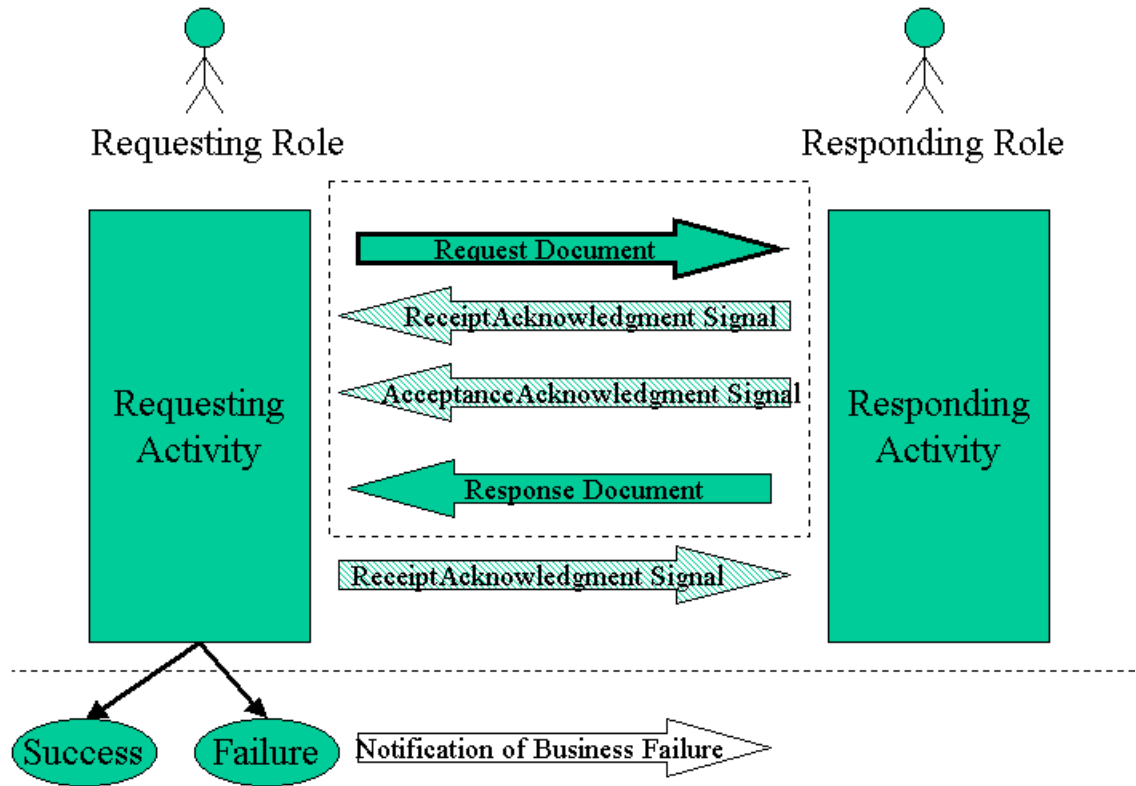
1134 The ebXML Business Transaction semantics allows you to specify electronic
1135 commerce transactions that provide

- 1136 • Interaction Predictability, i.e. have clear roles, clear transaction scope,
1137 clear time bounds, clear business information semantics, clear
1138 determination of success or failure.
- 1139 • Ability to create Legally Binding Contracts, i.e. the ability to specify that
1140 Business Transactions may be agreed to bind the parties.
- 1141 • Nonrepudiation, i.e. may specify the keeping of artifacts to aid in legal
1142 enforceability.
- 1143 • Authorization Security, i.e. may be specified to require athorization of
1144 parties performing roles.
- 1145 • Document Security, i.e. may be specified to be authorized, authenticated,
1146 confidential, tamperproof.
- 1147 • Reliability, i.e. the ability to specify reliable delivery of Business
1148 Documents and signals.
- 1149 • Run time Business Transaction Semantics, i.e. the rules and
1150 configuration parameters required for Business Service Interface software
1151 to predictably and deterministically execute ebXML Business
1152 Transactions.

1153 Each of the above characteristics of ebXML Business Transaction semantics is
1154 discussed in detail below.

1155 **6.5.1 Interaction Predictability**

1156 All Business Transactions follow a very precisely prescribed flow, or a
1157 precisely defined subset there-of. The following is an overall illustration of
1158 this flow. It can be thought of as the state machine across the two
1159 business partners. The N090R9.1 chapter on the UMM metamodel has a
1160 detail state chart for each of the business partners.
1161



1162

1163

Figure 13: Schematic of core Business Transaction semantics.

1164

1165

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

1166

1167

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.

1168

1169

1170

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.

1171

1172

1173

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.

1174

1175

1176

1177

1178

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

1179

1180

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

- 1181 6. After the receipt of the Request document flow, the responding activity
1182 may send a receiptAcknowledgement signal and/or an
1183 acceptanceAcknowledgement signal to the requesting role.
- 1184 7. The responding role then enters a responding activity. During or upon
1185 completion of the responding activity zero or one response is sent.
- 1186 8. Control will be returned back to the requesting activity if either a
1187 receiptAcknowledgement and/or acceptanceAcknowledgement and/or a
1188 response is specified as required. A receiptAcknowledgement (if required)
1189 must always occur before an acceptanceAcknowledgement (if required),
1190 and an acceptanceAcknowledgement must always occur before a
1191 response (if required). Control is returned to the requesting activity based
1192 on the last required of these three (if any). If none required, control stays
1193 with the responding activity.
- 1194 9. All business transactions succeed or fail. Success or failure depends on:
- 1195 a. The receipt or non-receipt of the request, the response and/or
1196 business signals
- 1197 b. The occurrence of time-outs
- 1198 c. The occurrence of a business exception
- 1199 d. The occurrence of a control exception
- 1200 e. The interpretation of the received response and guard
1201 expressions on transitions to success or failure
- 1202 10. The determination of Business Transaction success or failure is
1203 established by the requesting party based on the above success or
1204 failure factors. Once success or failure is thus established, the Business
1205 Transaction is considered closed with respect to both parties.
- 1206 11. Upon receipt of a response the requesting activity may send a
1207 receiptAcknowledgement signal back to the responding role. This is
1208 merely a signal and does not pass control back to the responding activity,
1209 nor does it alter the successful or failed completion of the Business
1210 Transaction that was based on the receipt of the Response.
- 1211 12. Upon identifying a time-out or exception in the processing of a Business
1212 Transaction, and closing the transaction accordingly, the requesting party
1213 may send a notification of failure to the responding party. This is
1214 considered a new Business Transaction and does not alter the already
1215 established conclusion of the Business Transaction.

1216

1217 6.5.1.1 Transaction Interaction Patterns

1218

1219

1220

1221

1222

1223

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.

1224 Furthermore, the specification of a business transaction may
1225 indicate, for the request whether receiptAcknowledgement and/or
1226 acceptanceAcknowledgement are required, and for the response
1227 whether receiptAcknowledgement is required.

1228 The way to specify that a receiptAcknowledgement is required is
1229 to set the parameter timeToAcknowledgeReceipt to any proper
1230 time duration other than zero. If this parameter has been set to a
1231 proper nonzero time duration, optionally either or both of the
1232 isIntelligibleCheckRequired and
1233 isNonrepudiationOfReceiptRequired parameters may also be set
1234 to 'Yes'.

1235 The way to specify that a acceptanceAcknowledgement is
1236 required is to set the parameter timeToAcknowledgeAcceptance
1237 to any proper time duration other than zero.

1238 So these two acknowledgement related parameters double as
1239 Boolean flags for whether the signal is required as part of the
1240 transaction, and as values for time-out of the transaction if the
1241 signal is not received.

1242 The specification of a business transaction may require each one
1243 of these signals independently of whether the other is required. If
1244 one is not required, it is actually not allowed. Therefore there is a
1245 finite set of combinations. The UMM supplies an illustrative set of
1246 patterns representing those combinations, for potential re-use.

1247

1248 6.5.2 Creating legally binding contracts

1249

1250 Trading partners may wish to indicate that a Business Transaction
1251 performed as part of an ebXML arrangement is, or is not, intended to be
1252 binding. A declaration of intent to be bound is a key element in
1253 establishing the legal equivalence of an electronic message to an
1254 enforceable signed physical writing. Parties may create explicit evidence
1255 of that intent by (1) adopting the ebXML Business Process Specification
1256 Schema standard and (2) manipulating the parameter ("isLegallyBinding")
1257 designated by the standard to indicate that intent.

1258

1259 In some early electronic applications, trading partners have simply used
1260 the presence, or absence, of an electronic signature (such as under the
1261 XML-DSIG standard) to indicate that intent. However, documents which
1262 rely solely on the presence of a signature may or may not be correctly
1263 interpreted, if there is semantic content indicating that a so-called
1264 contract is a draft, or nonbinding, or the like.

1265

1266 In ebXML, the presence or absence of an electronic signature cannot
1267 indicate by itself determine legally binding assent, because XML-DSIG
1268 signatures are reserved for other uses as an assurance of sender identity
1269 and message integrity.

1269

1270 isLegallyBinding is a parameter at the BusinessTransactionActivity level,
1271 which means that the performing of a BusinessTransaction within a
1272 Binary Collaboration is either specified as legally binding or not.
1273
1274 When operating under this standard, parties form binding agreements by
1275 exchanging binding messages that agree to terms (e.g., offer and
1276 acceptance).
1277 The "isLegallyBinding" parameter is Boolean, and its default value is
1278 "true." Under this standard, the exclusive manner for indicating that a
1279 Business Activity is not intended to be binding is to include a "false"
1280 value for the "isLegallyBinding" parameter for the transaction activity. As
1281 in EDI, the ebXML standard assumes that Business Transactions are
1282 intended by the trading parties to be binding unless otherwise indicated.
1283
1284 As a non-normative matter, parties may wish to conduct nonbinding
1285 transactions for a variety of reasons, including testing, and the exchange
1286 of proposed offers and counteroffers on a non-committal basis so as to
1287 discover a possible agreed set of terms. When using tangible signed
1288 documents, parties often do so by withholding a manual signature, or
1289 using a "DRAFT" stamp. In ebXML, trading partners may indicate that
1290 result by use of the "isLegallyBinding" parameter. See the illustrative
1291 Simple Negotiation Pattern set forth in the ebXML E-Commerce Patterns.

1292 6.5.3 Non-Repudiation

1293
1294 Trading partners may wish to conduct legally enforceable
1295 business transactions over ebXML. A party may elect to use non-
1296 repudiation protocols in order to generate documentation that
1297 would assist in the enforcement of the contractual obligation in
1298 court, in the case that the counterparty later attempts to repudiate
1299 its ebXML Business Documents and messages.

1300 Repudiation generally refers to the ability of a trading partner to
1301 argue at a later time, based on the persistent artifacts of a
1302 transaction, that it did not agree to the transaction. That argument
1303 might be based on assertions that a replying document was not
1304 sent, or was not sent by the proper party, or was incorrectly
1305 interpreted (under the applicable standard or the trading partners'
1306 business rules) as forming agreement.

1307 There are two kinds of non-repudiation protocol available under
1308 this document. Each protocol provides the user with some degree
1309 of additional evidentiary assurance by creating or requesting
1310 additional artifacts that would assist in a later dispute over
1311 repudiation issues. Neither is a dispositive absolute assurance.
1312 As in the paper world, trading partners are always free to invent
1313 colorful new arguments than an apparently-enforceable statement
1314 should be ignored. These parameters simply offer some
1315 opportunities to make that more difficult.

1316 One imposes a duty on each party to save copies of all Business
 1317 Documents and Document Envelopes comprising the transaction,
 1318 each on their own side, i.e., requestor saves his request,
 1319 responder saves his response. This is the
 1320 isNonRepudiationRequired parameter in the requesting or
 1321 responding activity. It is logically equivalent to a request that the
 1322 other trading partner maintain an audit trail. However, failure to
 1323 comply with that request is not necessarily computationally
 1324 detectable at run time, nor would it override the determination of a
 1325 "success" or "failure" end state.

1326 The other requires the responder to send a signed copy of the
 1327 receipt, which the requestor then saves. This is the
 1328 isNonRepudiationOfReceiptRequired parameter in the requesting
 1329 business activity.

1330 NonRepudiationOfReceipt is tied to the ReceiptAcknowledgement,
 1331 in that it requires the latter to be digitally signed. So
 1332 NonRepudiationOfReceipt is meaningless if
 1333 ReceiptAcknowledgement is not required. Failure to comply with
 1334 NonRepudiation of Receipt would be computationally detectable
 1335 at run time, and would override the determination of a "failure" end
 1336 state. If a timeToAcknowledgeReceipt is imposed on a
 1337 requesting message, and NonRepudiationOfReceipt is true, only
 1338 a digitally signed receipt will satisfy the imposed timeout deadline.
 1339 Thus, a failure to send a *signed* receipt within
 1340 timeToAcknowledgeReceipt, would make the transaction null and
 1341 void.

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

1342

1343 6.5.4 Authorization security

1344

1345 Each request or response may be sent by a variety of individuals,
 1346 representatives or automated systems associated with a business
 1347 partner. There may be cases where trading partners have more
 1348 than one ebXML-capable business service interface, representing
 1349 different levels of authority. In such a case, the parties may
 1350 establish rules regarding which interfaces or authors may be
 1351 confidently relied upon as speaking for the enterprise.

1352 In order to invoke those rules, a party may specify
 1353 IsAuthorizationRequired on a requesting or and responding
 1354 activity accordingly, with the result that [the activity] will only be

1355 processed as valid if the party interpreting it successfully matches
 1356 the stated identity of the activity's [Authorized Role] to a list of
 1357 allowed values previously supplied by that party.

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

1358

1359 IsAuthorizationRequired is specified on the requesting and
 1360 responding activity accordingly.

1361

1362 6.5.5 Document security

1363 The following security characteristics of each Business Document
 1364 being transported, even if many are collected in the same
 1365 message, can be specified individually, or collectively within a
 1366 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.

1367

1368 The value of *isConfidential*, *isTamperProof*, *isAuthenticated* at the
 1369 Document Envelope always applies to the primary Business
 1370 Document. It also applies to each of the attachments unless
 1371 specifically overridden at the Attachment level.

1372 When set to YES (or TRUE) these parameters assume that the
 1373 corresponding security characteristic is provided in a manner
 1374 providing persistence. Compliance requires that the specified
 1375 character of the document survive its reception at a business
 1376 service interface, and persist as the document is archived or
 1377 forwarded.

1378

1379 6.5.6 Reliability

1380

1381

1382

1383

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

1384

1385

1386

1387

1388

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.

1389 6.5.7 Parameters required for CPP/CPA

1390

1391

1392

1393

1394

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.

1395

1396

1397

1398

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.

1399

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

1400

1401

1402

1403

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.

1404

1405

1406

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.

1407 **6.6 Run time Business Transaction semantics**

1408

1409

1410

1411

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.

1412 Therefore, the Business Service Interface (BSI), or any software that implements
1413 one role in an ebXML collaboration needs at minimum to be able to support the
1414 following transaction semantics:

- 1415 1. Detection of the opening of a transaction
- 1416 2. Detection of transfer of control
- 1417 3. Detection of successful completion of a transaction
 - 1418 a. Application of business rules expressed as isPositiveResponse
 - 1419 and transition conditionGuard for determination of success
- 1420 4. Detection of failed completion of a transaction
 - 1421 a. Detection of time-outs
 - 1422 b. Detection of exceptions
 - 1423 c. Application of business rules expressed as isPositiveResponse
 - 1424 and transition conditionGuard for determination of failure
- 1425 5. Notification of failure
- 1426 6. Receipt of notification of failure
- 1427 7. Rollback upon failure (note this is the independent responsibility of each
1428 role, it is not a co-coordinated roll-back, there are no 2-phase commits in
1429 ebXML)

1430 ebXML does not specify how these transaction semantics are implemented but it
1431 is assumed that any Business Service Interface (BSI) will be able to support
1432 these basic transaction semantics at runtime. If either party cannot provide full
1433 support, then the requirements may be relaxed as overrides in the CPP/CPA.

1434 The following sections discuss the two causes of failure: Time-outs and
1435 Exceptions. When either one happens, it is the responsibility of the two roles to
1436 do the necessary roll-back, and to exit the transaction. The responsibilities of the
1437 two roles differ slightly and are described in each of the sections below.
1438 Generally, if a failure happens at the responding role, the responding role will
1439 send an exception signal to the requesting role, and both parties will exit the
1440 current transaction. If a failure happens at the requesting role, the requesting role
1441 will exit the current transaction and in a separate transaction notify the
1442 responding role about the failure. This way the flow of control within a transaction
1443 is always unambiguous and finite.

1444

1445 6.6.1 Timeouts

1446

1447 Since all business transactions must have a distinct time boundary, there
1448 are time-out parameters associated with the response, and each of the
1449 acknowledgement signals. If the time-out occurs before the
1450 corresponding response or signal arrives, the transaction is null and void.

1451

1452 Here are the time-out parameters relative to the three response types:

1453

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.

1454

1455

1456

1457

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.

1458

1459

1460

1461

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.

1462

1463

1464

1465

A BSI needs to comply with the above parameters to detect the appropriate time outs. To preserve the atomic semantics of the Business Transaction, the requesting and responding roles take different action based on time outs.

1466 A responding partner simply terminates if a timeout is thrown. This
1467 prevents responding business transactions from hanging indefinitely.

1468 A requesting partner terminates if a timeout is thrown and then sends a
1469 notification of failure to the responder as part of a separate transaction.

1470 When the time to perform an activity equals the time to acknowledge
1471 receipt or the time to acknowledge business acceptance then the highest
1472 priority time out exception must be used when the originator provides a
1473 reason for revoking their original business document offer. The time to
1474 perform exception is lower priority than both the time to acknowledge
1475 receipt and the time to acknowledge business acceptance.

1476

1477 6.6.2 Exceptions

1478

1479 Under all normal circumstances the response message and/or the time-
1480 outs determine the success or failure of a business transaction. However
1481 the business processing of the transaction can go wrong at either the
1482 responding or the requesting role.

1483 6.6.2.1 ControlException

1484

1485 A *ControlException* signals an error condition in the management of a
1486 business transaction. This business signal is asynchronously returned to
1487 the initiating activity that originated the request. This exception must
1488 terminate the business transaction. These errors deal with the
1489 mechanisms of message exchange such as verification, validation,
1490 authentication and authorization and will occur up to message
1491 acceptance. Typically the rules and constraints applied to the message
1492 will have only dealt with structure, syntax and message element values.

1493 6.6.2.2 Business Protocol Exceptions

1494

1495 A Business Protocol Exception (or *ProcessException*) signals an error
1496 condition in a business activity. This business signal is asynchronously
1497 returned to the initiating role that originated the request. This exception
1498 must terminate the *business transaction*. These errors deal with the
1499 mechanisms that process the *business transaction* and will occur after
1500 message verification and validation. Typically the rules and constraints
1501 applied to the message will deal with the semantics of message elements
1502 and the validity of the request itself. The content is not valid with respect to
1503 a responding role's business rules. This type of exception is usually
1504 generated after an *AcceptanceAcknowledgement* has been returned.

1505 A business protocol exception terminates the business transaction. The
1506 following are business protocol exceptions.

- 1507 • Negative acknowledgement of receipt. The structure/schema of a
1508 message is invalid.

- 1509 • Negative acknowledgement of acceptance. The business rules
1510 are violated.
- 1511 • Performance exceptions. The requested business action cannot
1512 be performed.
- 1513 • Sequence exceptions. The order or type of a business document
1514 or business signal is incorrect.
- 1515 • Syntax exceptions. There is invalid punctuation, vocabulary or
1516 grammar in the business document or business signal.
- 1517 • Authorization exceptions. Roles are not authorized to participate in
1518 the business transaction.
- 1519 • Business process control exceptions. Business documents are not
1520 signed for non-repudiation when required.

1521 A Business Transaction is defined in very atomic and deterministic terms.
1522 It always is initiated by the requesting role, and will always conclude at
1523 the requesting role. Upon receipt of the required response and/or signals,
1524 or time-out of same, the requesting role can unambiguously determine
1525 the success or failure of the Business Transaction. To preserve this
1526 semantics, control failures and business failures are treated differently by
1527 the requesting and responding roles as follows:

1528 A responding role that encounters a business protocol exception signals
1529 the exception back to the requesting role and then terminates the
1530 business transaction. If any business exceptions (includes negative
1531 receipt and acceptance acknowledgements) are signaled then the
1532 business transaction must terminate.

1533 A requesting role that encounters a business protocol exception
1534 terminates the transaction but does NOT send a business exception
1535 signal to the responding role. Rather, the requesting role then sends as a
1536 separate Business Transaction a notification revoking the offending
1537 business document request. This new transaction may be defined as a
1538 continuation of the current Binary Collaboration, or it may start a new
1539 Binary Collaboration specifically defined to handle this notification of
1540 failure.

1541 A BSI needs to comply specifically with the following parameters to
1542 produce the associated special exceptions. The requesting and
1543 responding roles take different action as per below.

1544 ***IsAuthorizationRequired***

1545 If a partner role needs authorization to request a business action
1546 or to respond to a business action then the sending partner role
1547 must sign the business document exchanged and the receiving
1548 partner role must validate this business control and approve the
1549 authorizer. A responding partner must signal an authorization
1550 exception if the requesting partner role is not authorized to
1551 perform the business activity. A sending partner must send

1552 notification of failed authorization if a requesting partner is not
1553 authorized to perform the responding business activity.

1554 ***IsNonRepudiationRequired***

1555 If non-repudiation of origin and content is required then the
1556 business activity must store the business document in its original
1557 form for the duration mutually agreed to in a trading partner
1558 agreement. A responding partner must signal a business control
1559 exception if the sending partner role has not properly delivered
1560 their business document. A requesting partner must send
1561 notification of failed business control if a responding partner has
1562 not properly delivered their business document.

1563 ***isNonRepudiationOfReceiptRequired.***

1564 Both partners agree to mutually verify receipt of a requesting
1565 business document and that the receipt must be non-repudiatable.
1566 A requesting partner must send notification of failed business
1567 control (possibly revoking a contractual offer) if a responding
1568 partner has not properly delivered their business document. For a
1569 further discussion of nonrepudiation of receipt, see also the
1570 ebXML E-Commerce and Simple Negotiation Patterns.

1571

1572 Non-repudiation of receipt provides the data for the following audit
1573 controls.

1574 **Verify responding role identity** (authenticate) – Verify the
1575 identity of the responding role (individual or organization) that
1576 received the requesting business document.

1577 **Verify content integrity** – Verify the integrity of the original
1578 content of the business document request.

1579 ***isPositiveResponse***

1580 An expression whose evaluation results in TRUE or FALSE. If
1581 TRUE this DocumentEnvelope is intended as a positive response
1582 to the request. The value for this parameter supplied for a
1583 DocumentEnvelope is an assertion by the sender of the
1584 DocumentEnvelope regarding its intent for the transaction to
1585 which it relates, but does not bind the recipient, or override the
1586 computation of transactional success or failure using the
1587 transaction's guard expressions.

1588 If a requesting role, upon evaluation of these expressions,
1589 determines a failure, then the requesting role will “roll back” the
1590 Business Transaction and send a notification of failure.

1591

1592

1593 **6.7 Runtime Collaboration Semantics**

1594 The ebXML collaboration semantics contain a number of relationships between
1595 multiparty collaborations and binary collaborations, between recursive layers of
1596 binary collaborations, and choreographies among transactions in binary
1597 collaborations. It is anticipated that over time BSI software will evolve to the point
1598 of monitoring and managing the state of a collaboration, similar to the way a BSI
1599 today is expected to manage the state of a transaction. For the immediate future,
1600 such capabilities are not expected and not required.

1601 **6.8 Where the ebXML Business Process Specification Schema 1602 May Be Implemented**

1603 The ebXML *Business Process Specification Schema* should be used wherever
1604 software is being specified to perform a role in an ebXML business collaboration.
1605 Specifically, the ebXML *Business Process Specification Schema* is intended to
1606 provide the business process and document specification for the formation of
1607 ebXML trading partner Collaboration Protocol Profiles and Agreements.

1608 However, the ebXML *Business Process Specification Schema* may be used to
1609 specify any electronic commerce collaboration. It may also be used for non-
1610 commerce collaborations, for instance in defining transactional collaborations
1611 among non-profit organizations or internally in enterprises.

1612 **7 UML Element Specification**

1613

1614 In the following we will review all the specification elements in the UML version of
1615 the ebXML *Business Process Specification Schema*, grouped as follows:

- 1616 • Business Collaborations
 - 1617 ○ Multiparty
 - 1618 ○ Binary
- 1619 • Business Transactions
- 1620 • Document flow
- 1621 • Choreography

1622

1623 **7.1 Business Collaborations**

1624

1625 **7.1.1 MultiPartyCollaboration**

1626 A Multiparty Collaboration is a synthesis of Binary Collaborations.
1627 A Multiparty Collaboration consists of a number of Business
1628 Partner Roles each playing roles in binary collaborations with
1629 each other.

1630	Tagged Values:	
1631	<i>name.</i>	Defines the name of the MultiPartyCollaboration
1632	Associations:	
1633	<i>partners</i>	A multiparty collaboration has two or more BusinessPartnerRoles
1634		
1635	Wellformedness Rules:	
1636		All multiparty collaborations must be synthesized from binary collaborations
1637		

1638

1639 7.1.2 BusinessPartnerRole

1640 A BusinessPartnerRole is the role played by a business partner in
 1641 a MultiPartyCollaboration. A BusinessPartnerRole performs at
 1642 most one Authorized Role in each of the Binary Collaborations
 1643 that make up the Multiparty Collaboration.

1644	Tagged Values:	
1645	<i>name.</i>	Defines the name of the role played by partner in the overall multiparty business collaboration, e.g. customer or supplier.
1646		
1647		

1648	Associations:	
1649	<i>performers.</i>	The Authorized Roles performed by a partner in the binary business collaboration.
1650		
1651	<i>transitions</i>	The transitions (managed by this BusinessPartnerRole) between activities across binary collaborations
1652		
1653		
1654	<i>collaboration</i>	The Business Partner Role participates in one multi party collaboration
1655		

1656	Wellformedness Rules:	
1657		A partner must not perform both roles in a given business activity.
1658		
1659		

1660 7.1.3 Performs

1661 Performs is an explicit modeling of the relationship between a
 1662 BusinessPartnerRole and the Roles it plays. This specifies the use
 1663 of an Authorized Role within a multiparty collaboration.

1664	Tagged Values:	
1665		
1666	<i>NONE</i>	

1667

Associations:

1668

performedBy An instance of Performs is performed by only one BusinessPartnerRole

1669

1670

authorizedRole The AuthorizedRole that will be performed by the Business PartnerRole

1671

1672

Wellformedness Rules:

1673

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

1674

1675

1676

1677

1678 **7.1.4 AuthorizedRole**

1679

1680

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.

1681

1682

1683

1684

Tagged Values:

1685

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

1686

1687

isInitiator Boolean, determining whether this authorized role is the initiator of its associated binary collaboration

1688

1689

1690

Associations:

1691

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

1692

1693

1694

from An AuthorizedRole may be the initiator in a business activity

1695

1696

to An AuthorizedRole may be the responder in a business activity

1697

1698

collaboration An AuthorizedRole may be in only one BinaryCollaboration

1699

1700

Wellformedness Rules:

1701

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

1702

1703

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

1704

1705

1706 7.1.5 BinaryCollaboration

1707 A Binary Collaboration defines a protocol of interaction between
1708 two authorized roles.

1709 A Binary Collaboration is a choreographed set of states among
1710 collaboration roles. The activities of performing business
1711 transactions or other collaborations are a kind of state.

1712 A Binary Collaboration choreographs one or more business
1713 transaction activities between two roles.

1714 A Binary Collaboration is not an atomic transaction and should not
1715 be used in cases where Business Transaction rollback is required.

1716 **Tagged Values:**

1717	<i>name</i>	Defines the name of the BinaryCollaboration
1718	<i>timeToPerform</i>	The period of time, starting upon initiation of the
1719		first activity, within which this entire collaboration
1720		must conclude.
1721	<i>preCondition</i>	A description of a state external to this
1722		collaboration that is required before this
1723		collaboration can commence.
1724	<i>postCondition</i>	A description of a state that does not exist
1725		before the execution of this collaboration but will
1726		exist as a result of the execution of this
1727		collaboration.
1728	<i>beginsWhen</i>	A description of an event external to the
1729		collaboration that normally causes this
1730		collaboration to commence.
1731	<i>endsWhen</i>	A description of an event external to this
1732		collaboration that normally causes this
1733		collaboration to conclude.
1734	<i>pattern</i>	The optional reference to a pattern that this
1735		binary collaboration is based on

1736 **Associations:**

1737	<i>role</i>	A binary collaboration consists of two authorized
1738		roles. One must be designated the Initiating
1739		Role, and one the Responding Role.
1740	<i>states</i>	A binary collaboration consists of one or more
1741		states, some of which are 'static', and some of
1742		which are action states
1743	<i>usedBy</i>	A binary collaboration may be used within
1744		another binary collaboration via a collaboration
1745		activity
1746	<i>transitions</i>	The transitions between activities in this binary
1747		collaboration

1748	Wellformedness Rules:	
1749	NONE	
1750		
1751	7.1.6 BusinessActivity	
1752		
1753	A business activity is an action state within a binary collaboration.	
1754	It is the super type for BusinessTransactionActivity and	
1755	CollaborationActivity, specifying the activity of performing a	
1756	transaction or another binary collaboration respectively.	
1757	Supertype of:	
1758	BusinessTransactionActivity, CollaborationActivity	
1759	Subtype of:	
1760	BusinessState	
1761	Tagged Values:	
1762	<i>name</i>	Defines the name of the activity uniquely within
1763		the binary collaboration
1764	Associations:	
1765	<i>from</i>	This must match one of the AuthorizedRoles in
1766		the parent binary collaboration and will become
1767		the initiator in the BinaryCollaboration performed
1768		by this activity
1769	<i>to</i>	This must match one of the AuthorizedRoles in
1770		the parent binary collaboration and will become
1771		the responder in the BinaryCollaboration
1772		performed by this activity
1773	Wellformedness Rules:	
1774	NONE	
1775		
1776	7.1.7 BusinessTransactionActivity	
1777	A business transaction activity defines the use of a business	
1778	transaction within a binary collaboration.	
1779	A business transaction activity is a business activity that executes	
1780	a specified business transaction. More than one instance of the	
1781	same business transaction activity can be open at one time if the	
1782	<i>isConcurrent</i> property is <i>true</i> .	
1783	Subtype of:	
1784	BusinessActivity	

1785	Tagged Values:	
1786	<i>timeToPerform</i>	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.
1787		
1788		
1789		
1790	<i>isConcurrent.</i>	If the BusinessTransactionActivity is concurrent then more than one instance of the associated BusinessTransaction can be performed as part of the execution of this BusinessTransactionActivity
1791		
1792		
1793		
1794		
1795	<i>isLegallyBinding</i>	Defines whether the Business Transaction performed by this activity is intended by the trading parties to be binding. Default value is True.
1796		
1797		
1798		

1799	Associations:	
1800	<i>uses.</i>	The business transaction activity performs (uses) exactly one business transaction.
1801		

1802	Wellformedness Rules:	
1803	NONE	
1804		

1805 7.1.8 CollaborationActivity

1806 A collaboration activity is the activity of performing a binary
1807 collaboration within another binary collaboration.

1808	Subtype of:	
1809	BusinessActivity	

1810	Tagged Values:	
1811	<i>NONE (other than inherited)</i>	

1812	Associations:	
1813	<i>uses</i>	A collaboration activity uses exactly one binary collaboration
1814		

1815	Wellformedness Rules:	
1816	A binary collaboration may not re-use itself	
1817		
1818		

1819 7.2 Business Transactions

1820

1821 7.2.1 BusinessTransaction

1822 A business transaction is a set of business information and
 1823 business signal exchanges amongst two commercial partners that
 1824 must occur in an agreed format, sequence and time period. If any
 1825 of the agreements are violated then the transaction is terminated
 1826 and all business information and business signal exchanges must
 1827 be discarded. Business Transactions can be formal as in the
 1828 formation of on-line offer/acceptance commercial contracts and
 1829 informal as in the distribution of product announcements.

1830 **Tagged Values:**

1831	<i>name</i>	Defines the name of the Business Transaction.
1832	<i>isGuaranteedDeliveryRequired</i>	Both partners must agree to use
1833		a transport that guarantees delivery
1834	<i>preCondition</i>	A description of a state external to this
1835		transaction that is required before this
1836		transaction can commence.
1837	<i>postCondition</i>	A description of a state that does not exist
1838		before the execution of this transaction but will
1839		exist as a result of the execution of this
1840		transaction.
1841	<i>beginsWhen</i>	A description of an event external to the
1842		transaction that normally causes this transaction
1843		to commence.
1844	<i>endsWhen</i>	A description of an event external to this
1845		transaction that normally causes this transaction
1846		to conclude.
1847	<i>pattern</i>	The optional reference to a pattern that this
1848		transaction is based on.

1849 **Associations:**

1850	<i>activities</i>	A BusinessTransaction can be performed by
1851		many BusinessTransactionActivites
1852	<i>requester</i>	A BusinessTransaction has exactly one
1853		RequestingBusinessActivity
1854	<i>responder</i>	A BusinessTransaction has exactly one
1855		RespondingBusinessActivity

1856 **Wellformedness Rules:**

1857 NONE

1858

1859

1860 7.2.2 Business Action

1861 A Business Action is an abstract super class. Business Action, is
 1862 the holder of attributes that are common to both Requesting
 1863 Business Activity and Responding Business Activity.

1864 **Supertype of:**

1865 RequestingBusinessActivity, RespondingBusinessActivity

1866 **Tagged Values:**

1867 *name* Defines the name of the
 1868 RequestingBusinessTransaction or
 1869 RespondingBusinessTransaction depending on
 1870 the subtype

1871 *IsAuthorizationRequired* Receiving party must validate identity
 1872 of originator against a list of authorized
 1873 originators. This parameter is specified on the
 1874 sending side. (See also section on action
 1875 security)

1876 *IsNonRepudiationRequired* Receiving party must check that
 1877 a requesting document is not garbled
 1878 (unreadable, unintelligible) before sending
 1879 acknowledgement of receipt. This parameter is
 1880 specified on the sending side. (See also section
 1881 on core transaction semantics)

1882 *isNonRepudiationOfReceiptRequired*. Requires the receiving
 1883 party to return a signed receipt, and the original
 1884 sender to save copy of the receipt. This
 1885 parameter is specified on the sending side.
 1886 (See also section on nonrepudiation)

1887 *timeToAcknowledgeReceipt* The time a receiving role has to
 1888 acknowledge receipt of a business document.
 1889 This parameter is specified on the sending side.
 1890 (See also section on core transaction semantics)

1891 *isIntelligibleCheckRequired* Receiving party must check that
 1892 a requesting document is not garbled
 1893 (unreadable, unintelligible) before sending
 1894 acknowledgement of receipt. This parameter is
 1895 specified on the sending side. (See also section
 1896 on core transaction semantics)

1897 **Associations:**

1898 NONE

1899 **Wellformedness Rules:**

1900 NONE

1901

1902

1903 **7.2.3 RequestingBusinessActivity**

1904 A RequestingBusinessActivity is a Business Action that is
 1905 performed by the requesting role within a Business Transaction. It
 1906 specifies the Document Envelope which will carry the request.

1907

Subtype of:

1908

BusinessAction

1909

Tagged Values:

1910

1911

1912

1913

1914

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)

1915

Associations:

1916

1917

transaction A requesting activity is performed in exactly one
 business transaction

1918

1919

documentEnvelope A requesting activity sends exactly one
 Document Envelope

1920

1921

Wellformedness Rules:

1922

NONE

1923

1924 **7.2.4 RespondingBusinessActivity**

1925

1926

1927

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.

1928

1929

1930

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

1931

Subtype of:

1932

BusinessAction

1933

Tagged Values:

1934

NONE, except as inherited from Business Action

1935

Associations:

1936

1937

transaction A responding activity is performed in exactly one
 business transaction

1938 *DocumentEnvelope* A responding activity may specify zero
 1939 or more but sends at most one Document
 1940 Envelope

1941

1942 **Wellformedness Rules:**

1943 NONE

1944

1945 **7.3 Document flow**

1946

1947 7.3.1 Document Security

1948 DocumentSecurity is an abstract super class holding the security
 1949 related attributes for DocumentEnvelope and Attachment.

1950 **Supertype of:**

1951 DocumentEnvelope and Attachment

1952 **Tagged Values:**

1953 *IsAuthenticated* There is a digital certificate associated with the
 1954 document entity. This provides proof of the
 1955 signer's identity. (See also section on Document
 1956 Security)

1957 *IsConfidential* The information entity is encrypted so that
 1958 unauthorized parties cannot view the
 1959 information. (See also section on Document
 1960 Security)

1961 *isTamperProof* The information entity has an encrypted
 1962 message digest that can be used to check if the
 1963 message has been tampered with. This requires
 1964 a digital signature (sender's digital certificate
 1965 and encrypted message digest) associated with
 1966 the document entity. (See also section on
 1967 Document Security)

1968 **Associations:**

1969 NONE

1970 **Wellformedness Rules:**

1971 NONE

1972 7.3.2 Document Envelope

1973 A Document Envelope is what conveys business information
 1974 between the two roles in a business transaction. One Document
 1975 Envelope conveys the request from the requesting role to the
 1976 responding role, and another Document Envelope conveys the

1977		response (if any) from the responding role back to the requesting
1978		role.
1979	Subtype of:	
1980		DocumentSecurity
1981		
1982	Tagged Values:	
1983		<i>isPositiveResponse</i> TRUE or FALSE. If TRUE this
1984		DocumentEnvelope is intended as a positive
1985		response to the request. This parameter is only
1986		relevant on the response envelope. Its value
1987		does not bind the recipient, or override the
1988		computation of transactional success or failure
1989		using the transaction's guard expressions.
1990	Associations:	
1991		<i>requesting</i> This is a reference to the requesting activity
1992		associated with this DocumentEnvelope. This
1993		requesting activity may be the sender, or the
1994		receiver depending on whether the
1995		DocumentEnvelope represents a request or a
1996		response.
1997		<i>responding</i> This is a reference to the requesting activity
1998		associated with this DocumentEnvelope. This
1999		responding activity may be the sender, or the
2000		receiver depending on whether the
2001		DocumentEnvelope represents a request or a
2002		response.
2003		<i>BusinessDocument</i> This identifies the primary Business
2004		Document in the envelope. A Document
2005		Envelope contains exactly one primary Business
2006		Document.
2007		<i>attachment</i> A Document Envelope contains an optional set
2008		of attachments related to the primary document
2009	Wellformedness Rules:	
2010		A Document Envelope is associated with exactly one requesting
2011		and one responding activity.
2012		IsPositiveResponse is not a relevant parameter on a
2013		DocumentEnvelope sent by a requesting activity
2014		.
2015		
2016		
2017		
2018		.

2019

2020 **7.3.3 BusinessDocument**

2021 BusinessDocument is a generic name of a document.

2022

Tagged Values:

2023

2024

2025

name Defines the generic name of the Business Document as it is known within this Business Process Specification

2026

2027

2028

2029

conditionExpression A Business Document may have one Condition Expression. This determines whether this is a valid business document for its envelope

2030

Associations:

2031

2032

documentEnvelope A Business Document can be in multiple Document Envelopes

2033

2034

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

2035

Wellformedness Rules:

2036

NONE

2037

2038 **7.3.4 Attachment**

2039

2040

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

2041

Subtype of:

2042

DocumentSecurity

2043

Tagged Values:

2044

name Defines the name of the attachment

2045

2046

mimeType Defines the valid MIME (Multipurpose Internet Mail Extensions) type of this Attachment

2047

2048

specification A reference to an external source of description of this attachment.

2049

version The version of the Attachment

2050

Associations:

2051

2052

documentEnvelope An Attachment is in exactly one Document Envelope

2053

2054

businessDocument An Attachment can be defined by a BusinessDocument. If it is not of a defined

2055 Business Document, the mime type and spec
2056 will be the only indication of its type.

2057 **Wellformedness Rules:**

2058 NONE

2059
2060

2061 **7.4 Choreography within Collaborations.**

2062
2063

2064 **7.4.1 BusinessState**

2065 A business state is any state that a binary collaboration can be in.
2066 Start and CompletionState are a snapshot right before or right
2067 after an activity, BusinessActivity is an action states that denote
2068 the state of being in an activity. Fork and Join reflect the activity of
2069 forking to multiple activities or joining back from them.

2070 **Supertype of:**

2071 Start, CompletionState, Fork, Join, BusinessActivity

2072 **Tagged Values:**

2073 *none*

2074 **Associations:**

2075 *collaboration* A business state belongs to only one binary
2076 collaboration

2077 *entering* A transition that reflects entry into this state

2078 *exiting* A transition that reflects exiting from this state

2079 **Wellformedness Rules:**

2080 NONE

2081

2082 **7.4.2 Transition**

2083 A transition is a transition between two business states in a binary
2084 collaboration.

2085 Choreography is expressed as transitions between business
2086 states

2087 **Tagged Values:**

2088 *onInitiation* This specifies this is a nested
2089 BusinessTransactionActivity and that upon

2090 receipt of the request in the associated
 2091 transaction a second activity is performed before
 2092 returning to the transaction to send the response
 2093 back to the original requestor.

2094 *conditionGuard* A reference to the status of the previous
 2095 transaction. A fixed value of Success,
 2096 BusinessFailure, TechnicalFailure, or AnyFailure

2097 *conditionExpression* A transition may have one Condition
 2098 Expression. For a transition, this determines
 2099 whether this transition should happen or not.

2100 **Associations:**

2101 *in* The business state this transition is entering
 2102 *out* The business state this transition is exiting

2103 **Wellformedness Rules:**

2104 A transition cannot enter and exit the same state

2105

2106 **7.4.3 Start**

2107 The starting state for a Binary Collaboration. A Binary
 2108 Collaboration should have at least one starting activity. If none
 2109 defined, then all activities are considered allowable entry points.

2110 **Subtype of:**

2111 BusinessState

2112 **Tagged Values:**

2113 NONE

2114 **Associations:**

2115 NONE

2116 **Wellformedness Rules:**

2117 NONE

2118 **7.4.4 CompletionState**

2119 The ending state of an binary collaboration, sub classed by
 2120 success and failure

2121 **Supertype of:**

2122 Success, Failure

2123 **Subtype of:**

2124 BusinessState

2125		Tagged Values:
2126		<i>NONE</i>
2127		Associations:
2128		<i>NONE</i>
2129		Wellformedness Rules:
2130		<i>NONE</i>
2131	7.4.5 Success	
2132		Defines the successful conclusion of a binary collaboration as a
2133		transition from an activity.
2134		Subtype of:
2135		CompletionState
2136		Tagged Values:
2137		<i>conditionExpression</i> A success state may have one
2138		Condition Expression for the transition. This
2139		determines whether this transition should
2140		happen or not.
2141		Associations:
2142		<i>NONE</i> , except as inherited
2143		Wellformedness Rules:
2144		Every activity Binary Collaboration should have at least one
2145		success
2146		
2147	7.4.6 Failure	
2148		A subtype of CompletionState which defines the unsuccessful
2149		conclusion of a binary collaboration as a transition from an activity.
2150		Subtype of:
2151		CompletionState
2152		Tagged Values:
2153		<i>conditionExpression</i> A failure state may have one Condition
2154		Expression for the transition. This determines
2155		whether this transition should happen or not.
2156		Associations:
2157		<i>NONE</i> , except as inherited

2158		Wellformedness Rules:	
2159		Every Binary Collaboration should have at least one failure	
2160	7.4.7 Fork		
2161		A Fork is a state with one inbound transition and multiple	
2162		outbound transitions. All activities pointed to by the outbound	
2163		transitions are assumed to happen in parallel.	
2164		Subtype of:	
2165		BusinessState	
2166		Tagged Values:	
2167		<i>Name</i>	Defines the name of the Fork state
2168		Associations:	
2169		<i>None</i>	
2170		Wellformedness Rules:	
2171		<i>None</i>	
2172			
2173	7.4.8 Join		
2174		A business state where an activity is waiting for the completion of	
2175		one or more other activities. Defines the point where previously	
2176		forked activities join up again.	
2177		Subtype of:	
2178		BusinessState	
2179		Tagged Values:	
2180		<i>Name</i>	Defines the name of the Join state
2181		<i>waitForAll</i>	Boolean value indicating if this Join state should
2182			wait for all incoming transitions to complete. If
2183			TRUE, wait for all, if False proceed on first
2184			incoming transition.
2185		Associations:	
2186		<i>None</i>	
2187		Wellformedness Rules:	
2188		<i>None</i>	
2189			
2190			
2191			

2192 **7.5 Definition and Scope**

2193 The ebXML *Business Process Specification Schema* should be used wherever
 2194 software is being specified to perform a role in an ebXML binary collaboration.
 2195 Specifically, the ebXML *Business Process Specification Schema* is intended to
 2196 provide the business process and document specification for the formation of a
 2197 trading partner Collaboration Protocol Profile and Agreement. A set of
 2198 specification rules have been established to properly constrain the expression of
 2199 a business process and information model in a way that can be directly
 2200 incorporated into a trading partner Collaboration Protocol Profile and Agreement.

2201 **7.6 Collaboration and transaction well-formedness rules**

2202 The following rules should be used in addition to standard parsing to properly
 2203 constrain the values of the attributes of the elements in an ebXML Business
 2204 Process Specification.

2205 *Business Transaction*

- 2206 [0] If non-repudiation is required then the input or returned business
 2207 document must be a tamper-proofed entity.
- 2208 [1] If authorization is required then the input business document and
 2209 business signal must be an authenticated or a tamper proofed secure
 2210 entity.
- 2211 [2] The time to acknowledge receipt must be less than the time to
 2212 acknowledge acceptance if both properties have values.
 2213
$$\text{timeToAcknowledgeReceipt} < \text{timeToAcknowledgeAcceptance}$$
- 2215 [3] If the time to acknowledge acceptance is null then the time to perform
 2216 an activity must either be equal to or greater than the time to
 2217 acknowledge receipt.
- 2218 [4] The time to perform a transaction cannot be null if either the time to
 2219 acknowledge receipt or the time to acknowledge acceptance is not
 2220 null.
- 2221 [5] If non-repudiation of receipt is required then the time to acknowledge
 2222 receipt cannot be null.
- 2223 [6] The time to acknowledge receipt, time to acknowledge acceptance
 2224 and time to perform cannot all be zero.
- 2225 [7] If non-repudiation is required at the requesting business activity, then
 2226 there must be a responding business document.

2227 *RequestingBusinessActivity*

- 2228 [8] There must be one input transition whose source state vertex is an
 2229 initial pseudo state.

- 2230 [9] There must be one output transition whose target state vertex is a
2231 final state specifying the state of the machine when the activity is
2232 successfully performed.
- 2233 [10] There must be one output transition whose target state vertex is a
2234 final state specifying the state of the machine when the activity is NOT
2235 successfully performed due to a process control exception.
- 2236 [11] There must be one output transition whose target state vertex is a
2237 final state specifying the state of the machine when the activity is NOT
2238 successfully performed due to a business process exception.
- 2239 [12] There must be one output document flow from a requesting
2240 business activity that in turn is the input to a responding business
2241 activity.
- 2242 [13] There must be zero or one output document flow from a
2243 responding business activity that in turn is the input to the requesting
2244 business activity.
- 2245 *RespondingBusinessActivity*
- 2246 [14] There must be one input transition from a document flow that in
2247 turn has one input transition from a requesting business activity.
- 2248 [15] There must be zero or one output transition to an document flow
2249 that in turn has an output transition to a requesting business activity.
- 2250 *Business Collaboration*
- 2251 [16] A Business Partner Role cannot provide both the initiating and
2252 responding roles of the same business transaction activity.

2253 **8 ebXML Business Process Specification Schema –** 2254 **(DTD)**

2255 In this section we describe the DTD and XML Schema version of the
2256 Specification Schema. There are minimal differences between the DTD and the
2257 XML Schema, therefore the elements will only be described once, noting
2258 differences when needed. This discussion includes

- 2259 • An example XML Business Process Specification listed in Appendix A.
- 2260 • A listing of the DTD in Appendix B and the XML Schema in Appendix C
- 2261 • A table listing all the elements with definitions and parent/child
2262 relationships
- 2263 • A table listing all the attributes with definitions and parent element
2264 relationships
- 2265 • A table listing all the elements, each with a cross reference to the
2266 corresponding class in the UML version of the specification schema
- 2267 • Rules about namespaces and element references

2268 **8.1 Documentation for the DTD**

2269 This section will document the DTD. The DTD has been derived from the UML
2270 model. The correlation between the UML classes and DTD elements will be
2271 shown separately later in this document.
2272

2273 Overall Structure excluding attribute definitions:

2274 [ProcessSpecification](#) (Documentation*, SubstitutionSet*, (Include* | BusinessDocument* |
2275 ProcessSpecification* | Package | BinaryCollaboration |
2276 BusinessTransaction | MultiPartyCollaboration*))

2277 [Documentation](#)()

2278 [Include](#)(Documentation*)

2279 BusinessDocument (ConditionExpression | Documentation)*

2280 ConditionExpression (Documentation*)

2281 SubstitutionSet (DocumentSubstitution | AttributeSubstitution | Documentation)*

2282 DocumentSubstitution (ConditionExpression | Documentation)*

2283 AttributeSubstitution (Documentation*)

2284 [Package](#)(Documentation*, (Package | BinaryCollaboration |

2285 BusinessTransaction | MultiPartyCollaboration)*)

2286 [BinaryCollaboration](#)(Documentation*, InitiatingRole, RespondingRole,

2287 (Documentation* | Start | Transition | Success | Failure |

2288 BusinessTransactionActivity | CollaborationActivity | Fork | Join)*)

2289 InitiatingRole (Documentation*)

2290 [RespondingRole](#)(Documentation*)

2291 [Start](#)(Documentation*)

2292 [Transition](#)(ConditionExpression | Documentation)*

2293 [Success](#)(ConditionExpression | Documentation)*

2294 [Failure](#)(ConditionExpression | Documentation)*

2295 [Fork](#)(Documentation*)
 2296 [Join](#)(Documentation*)
 2297 [BusinessTransactionActivity](#)(Documentation*)
 2298 [CollaborationActivity](#)(Documentation*)
 2299 [BusinessTransaction](#)(Documentation*, RequestingBusinessActivity,
 2300 RespondingBusinessActivity)
 2301 [RequestingBusinessActivity](#)(Documentation*, DocumentEnvelope)
 2302 [RespondingBusinessActivity](#)(Documentation*, DocumentEnvelope*)
 2303 [MultiPartyCollaboration](#)(Documentation*, BusinessPartnerRole*)
 2304 [BusinessPartnerRole](#)(Documentation*, Performs*, Transition*)
 2305 [Performs](#)(Documentation*)
 2306 [Transition](#)(Documentation*)

a. Attachment

2309 **XML Element Name:** Attachment

2310 **DTD Declaration:**

```
2311 <!ELEMENT Attachment (Documentation*)>
2312 <!ATTLIST Attachment
2313     name                CDATA #REQUIRED
2314     nameID              ID #IMPLIED
2315     BusinessDocument    CDATA #IMPLIED
2316     BusinessDocumentIDRef IDREF #IMPLIED
2317     mimeType            CDATA #IMPLIED
2318     specification       CDATA #IMPLIED
2319     version              CDATA #IMPLIED
2320     isConfidential       (true | false) "false"
2321     isTamperProof        (true | false) "false"
2322     isAuthenticated      (true | false) "false">
```

Definition:

2324 An optional attachment to a BusinessDocument in a DocumentEnvelope.

2325

2326 **Parent Elements:**

- 2327 • DocumentEnvelope

2328

2328

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's type 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

2329

2330

b. AttributeSubstitution

2331

Element Name: AttributeSubstitution

2332

DTD Declaration:

2333

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

2334

```
<!ATTLIST AttributeSubstitution
```

2335

```
attributeName CDATA #IMPLIED
```

2336

```
value CDATA #IMPLIED
```

2337

```
>
```

2338

Definition:

2339

Attribute Substitution specifies that an attribute value should be used in place of some attribute value in an existing process specification.

2340

2341

Parents:

2342

- SubstitutionSet

2343

Attributes:

2344

2345

Attribute Name	Definition	Default Value
attributeName	The name of an attribute of any element within the scope of the substitution set.	Required Input
value	The value which shall replace the current value of the attribute.	Required Input

2348

c. Binary Collaboration

2349

XML Element Name: BinaryCollaboration

2350

DTD Declaration:

2351

```
<!ELEMENT BinaryCollaboration (Documentation*,
```

2352

```
InitiatingRole, RespondingRole, (Documentation* | Start |
```

2353

```
Transition | Success | Failure |
```

2354

```
BusinessTransactionActivity | CollaborationActivity | Fork
```

2355

```
| Join)*>
```

2356

```
<!ATTLIST BinaryCollaboration
```

2357

```
name CDATA #REQUIRED
```

2358

```
nameID ID #IMPLIED
```

2359

```
pattern CDATA #IMPLIED
```

2360

```
beginsWhen CDATA #IMPLIED
```

2361

```
endsWhen CDATA #IMPLIED
```

2362

```
precondition CDATA #IMPLIED
```

2363

```
postCondition CDATA #IMPLIED
```

2364

```
timeToPerform CDATA #IMPLIED
```

2365

```
>
```

2366

Definition:

2367

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

2368

2369

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.

2370

2371

2372 A Binary Collaboration choreographs one or more business transaction activities
2373 between two roles.

2374 A Binary Collaboration is not an atomic transaction and should not be used in
2375 cases where Business Transaction rollback is required.

2376

2377

Parents:

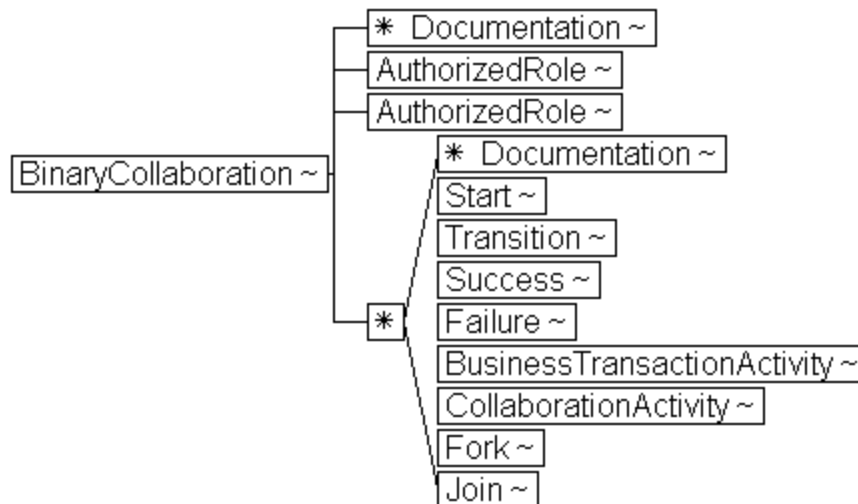
2378

- Package

2379

2380

Hierarchical Model:



2381

2382

Attributes:

2383

Attribute Name	Definition	Default Value
name	Defines the name of the Binary Collaboration.	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	Optional Input.

	collaboration can commence.	
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.

2384

2385

d. BusinessDocument

2386

Element Name: BusinessDocument

2387

DTD Declaration:

2388

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

2389

2390

```
<!ATTLIST BusinessDocument
name CDATA #REQUIRED
nameID ID #IMPLIED
specificationLocation CDATA #IMPLIED
specificationElement CDATA #IMPLIED>
```

2391

2392

2393

2394

2395

Definition:

2396

BusinessDocument is a generic name of a document.

2397

Parents:

2398

- Attachment

2399

2400

2401

Attributes:

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
specificationLocation	Reference to an external source of the schema definition. In the XML Schema version the data type is xsd:anyURI	Optional
specificationElement	Reference to the element within the schema definition that defines this document.	Optional

2402

2403

e. Business Partner Role

2404

2405

2406

Element Name: BusinessPartnerRole

2407

DTD Declaration:

2408

```
<!ELEMENT BusinessPartnerRole (Documentation*, Performs*,
2409                                     Transition*)>
```

2410

```
<!ATTLIST BusinessPartnerRole
2411     name    CDATA #REQUIRED
2412     nameID  ID    #IMPLIED>
```

2411

2412

2413

2414

Definition:

2415

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.

2416

2417

2418

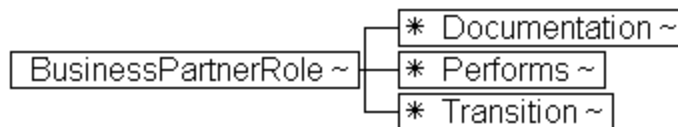
2419

Parents:

2420

- MultiPartyCollaboration

2421

Hierarchical Model:

2422

2423

2424

Attributes:

Attribute Name	Definition	Default Value
Name	Defines the name of the role played by a partner in the overall multiparty business collaboration, e.g. customer or supplier.	Required Input.
nameID	The XML ID version of name	Optional

2425

2426

f. Business Transaction

2427

Element Name: BusinessTransaction

2428

Content Model:

2429

```
<!ELEMENT BusinessTransaction (Documentation*,
2430 RequestingBusinessActivity, RespondingBusinessActivity)>
```

2431

```
<!ATTLIST BusinessTransaction
```

2432

```
    name    CDATA #REQUIRED
```

2433

```
    nameID  ID    #IMPLIED
```

2434

```
    pattern CDATA #IMPLIED
```

2435

```
    beginsWhen CDATA #IMPLIED
```

2436

```
    endsWhen CDATA #IMPLIED
```

2437

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

2438

```
    precondition CDATA #IMPLIED
```

2439

```
    postCondition CDATA #IMPLIED>
```

2440

2441

Definition:

2442 A business transaction is a set of business information and business
 2443 signal exchanges amongst two commercial partners that must occur in
 2444 an agreed format, sequence and time period. If any of the agreements
 2445 are violated then the transaction is terminated and all business
 2446 information and business signal exchanges must be discarded. Business
 2447 Transactions can be formal as in the formation of on-line
 2448 offer/acceptance commercial contracts and informal as in the distribution
 2449 of product announcements.

2450

2451

2452

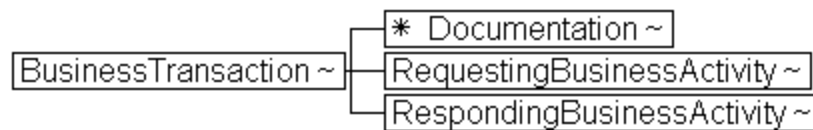
Parents:

2453

- Package

2454

2455

Hierarchical Model:

2456

2457

2458

Attributes:

2459

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:

	guarantees delivery	{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.

2460

2461

g. Business Transaction Activity

2462

Element Name: BusinessTransactionActivity

2463

Content Model:

2464

`<!ELEMENT BusinessTransactionActivity (Documentation*)>`

2465

`<!ATTLIST BusinessTransactionActivity`

2466

`name CDATA #REQUIRED`

2467

`nameID ID #IMPLIED`

2468

`businessTransaction CDATA #REQUIRED`

2469

`businessTransactionIDRef IDREF #IMPLIED`

2470

`fromAuthorizedRole CDATA #REQUIRED`

2471

`fromAuthorizedRoleIDRef IDREF #IMPLIED`

2472

`toAuthorizedRole CDATA #REQUIRED`

2473

`toAuthorizedRoleIDRef IDREF #IMPLIED`

2474

`isConcurrent (true | false) "false"`

2475

`isLegallyBinding (true | false) "true"`

2476

`timeToPerform CDATA #IMPLIED>`

2477

Definition:

2478

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

2479

2480

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.

2481

2482

2483

Parents:

2484

- BinaryCollaboration

2485

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	Required Input.

	Transaction Activity	
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}

2486

2487

h. Collaboration Activity

2488

Element Name: CollaborationActivity

2489

DTD Declaration:

```

2490 <!ELEMENT CollaborationActivity (Documentation*)>
2491 <!ATTLIST CollaborationActivity
2492     name                CDATA #REQUIRED
2493     nameID              ID #IMPLIED
2494     fromAuthorizedRole  CDATA #REQUIRED
2495     fromAuthorizedRoleIDRef CDATA #IMPLIED
2496     toAuthorizedRole    CDATA #REQUIRED
2497     toAuthorizedRoleIDRef CDATA #IMPLIED
2498     binaryCollaboration CDATA #REQUIRED>
2499     binaryCollaborationIDRef CDATA #IMPLIED>

```

Definition:

2501 A collaboration activity is the activity of performing a binary collaboration within
 2502 another binary collaboration.

2503

Parents:

2504

- BinaryCollaboration

2505

2506

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.

2507

2508

i. Documentation

2509

Element Name: Documentation

2510

DTD Declaration:

2511

<!ELEMENT Documentation (#PCDATA)>

2512

<!ATTLIST Documentation

2513

uri CDATA #IMPLIED>

2514

2515

Definition:

2516

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

2517

2518

2519

Parents:

2520

- AuthorizedRole

2521

- BinaryCollaboration

2522

- BusinessPartnerRole

2523

- BusinessTransaction

2524

- BusinessTransactionActivity

2525

- CollaborationActivity

2526

- DocumentEnvelope

2527

- BusinessDocument

2528

- ProcessSpecification

2529

- MultiPartyCollaboration

2530

- Package

2531

- Performs

2532

- RequestingBusinessActivity

2533

- RespondingBusinessActivity

2534

- Transition

2535

Attributes:

2536

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.

2537

2538

j. DocumentEnvelope

2539

Element Name: DocumentEnvelope

2540

Content Model:

```

2541 <!ELEMENT DocumentEnvelope (Documentation*,
2542 Attachment*)>
2543 <!ATTLIST DocumentEnvelope
2544 businessDocument CDATA #REQUIRED
2545 businessDocumentIDRef IDREF #IMPLIED
2546 isPositiveResponse CDATA #IMPLIED
2547 isAuthenticated (true | false) "false"
2548 isConfidential (true | false) "false"
2549 isTamperProof (true | false) "false">
2550
2551 Definition:

```

2552 A DocumentEnvelope is what conveys business information between the two
 2553 roles in a business transaction. One DocumentEnvelope conveys the request
 2554 from the requesting role to the responding role, and another DocumentEnvelope
 2555 conveys the response (if any) from the responding role back to the requesting
 2556 role.

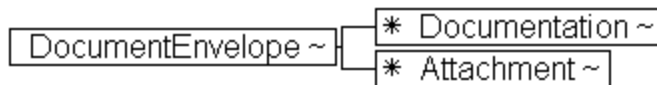
2557 **Parents:**

- 2558 • RequestingBusinessActivity
- 2559 • RespondingBusinessActivity

2560

2561

Hierarchical Model:



2562

2563

2564

Attributes:

Attribute Name	Definition	Default Value
businessDocument	The name of the business document.	Required Input.
businessDocumentIDRef	The XML IDREF version of businessDocument	Optional Input.
isPositiveResponse	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	Optional Input.

	<p>this could be an XPath expression that interrogates the BusinessDocument in the envelope.</p> <p>IsPositiveResponse is only relevant for responses, and is ignored in requests.</p>	
isAuthenticated	<p>There is a digital certificate associated with the document entity. This provides proof of the signer's identity.</p> <p>(See also section on Document Security)</p>	<p>false</p> <p>Valid Values: {true, false}</p>
isConfidential	<p>The information entity is encrypted so that unauthorized parties cannot view the information.</p> <p>(See also section on Document Security)</p>	<p>false</p> <p>Valid Values: {true, false}</p>
isTamperProof	<p>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.</p> <p>(See also section on Document Security)</p>	<p>false</p> <p>Valid Values: {true, false}</p>

2565

2566

2568

2570

2571

2572

k. DocumentSubstitution

2573

Element Name: DocumentSubstitution

2574

DTD Declaration:

2575

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

2576

```
<!ATTLIST DocumentSubstitution
```

2577

```
    originalBusinessDocument CDATA #IMPLIED
```

2578

```
    originalBusinessDocumentID IDREF #IMPLIED
```

2579

```
    substituteBusinessDocument CDATA #IMPLIED
```

2580

```
    substituteBusinessDocumentId IDREF #IMPLIED
```

2581

Definition:

2582 DocumentSubstitution specifies a document that should be used in place of a
2583 document in an existing process specification.

2584

Parents:

2585

- SubstitutionSet

2586

2587

2588

Attributes:

Attribute Name	Definition	Default Value
originalBusinessDocument	The name of a business document within the scope of the substitution set.	Required Input
originalBusinessDocumentID	The ID of the business document.	Optional
substituteBusinessDocument	The document which shall replace the current document.	Required Input
substituteBusinessDocumentID	The ID of the replacement document.	Optional

2589

I. Failure

2590

Element Name: Failure

2591

DTD Declaration:

2592

```
<!ELEMENT Failure (ConditionExpression, Documentation*)
```

2593

```
>
```

2594

```
<!ATTLIST Failure
```

2595

```
fromBusinessState CDATA #REQUIRED
```

2596

```
fromBusinessStateIDRef IDREF #IMPLIED
```

2597

```
conditionGuard (Success | BusinessFailure |
```

2598

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

2599

2600

Definition:

2601

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

2602

2603

Parents:

2604

- BinaryCollaboration

2605

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
conditionGuard	The condition that guards this transition	Optional Valid Values: {Success, BusinessFailure, TechnicalFailure, AnyFailure}

2606

2607

m. Fork

2608

Element Name: Fork

2609

DTD Declaration:

2610

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

2611

```
<!ATTLIST Fork
```

2612

```
    name    CDATA    #REQUIRED
```

2613

```
    nameID  ID      #IMPLIED >
```

2614

Definition:

2615

A Fork is a state with one inbound transition and multiple outbound transitions. All activities pointed to by the outbound transitions are assumed to happen in parallel.

2616

2617

2618

Parents:

2619

- BinaryCollaboration

2620

Attributes:

2621

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

2622

2623

n. Include

2624

Element Name: Include

2625

DTD Declaration:

2626

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

2627

```
<!ATTLIST Include
```

2628

```
    name    CDATA    #REQUIRED
```

2629

```
    version CDATA    #REQUIRED
```

2630

```
    uuid    CDATA    #REQUIRED
```

2631

```
    uri     CDATA    #REQUIRED >
```

2632

Definition:

2633

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.

2634

2635

2636

2637 Documents are merged based on name scope. A name in an included package
2638 will be indistinguishable from a name in the base document.

2639 **Parents:**

- ProcessSpecification

2641 **Hierarchical Model:**

* Include ~ * Documentation ~

2642

2643

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

2644

2645

2646

o. Initiating Role

2647

XML Element Name: InitiatingRole

2648

DTD Declaration:

2649

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

2650

```
<!ATTLIST InitiatingRole
```

2651

```
    name CDATA #REQUIRED
```

2652

```
    nameID ID #IMPLIED>
```

2653

2654

Definition:

2655

An Initiating Role is a role that is authorized to send the first request, e.g. the buyer is authorized to send the request for purchase order. The Initiating Role initiates the binary collaboration. Typically this is the first role.

2656

2657

2658

Parents:

2659

BinaryCollaboration

2660

2661

Attributes:

Attribute Name	Definition	Default Value
Name	Defines the name of the Initiating Role	Required input.

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

2662

p. Join

2663

Element Name: Join

2664

DTD Declaration:

2665

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

2666

```
<!ATTLIST Join
```

2667

```
  name          CDATA #REQUIRED
```

2668

```
  nameID       ID      #IMPLIED
```

2669

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

2670

Definition:

2671

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

2672

2673

Parents:

2674

- BinaryCollaboration

2675

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}

2676

2677

q. MultiParty Collaboration

2678

Element Name: MultiPartyCollaboration

2679

DTD Declaration:

2680

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

2681

2682

```
<!ATTLIST MultiPartyCollaboration
```

2683

```
  name          CDATA #REQUIRED
```

2684

```
  nameID       ID      #IMPLIED >
```

2685

Definition:

2686

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.

2687

2688

2689

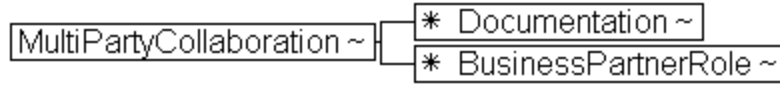
Parents:

2690

- Package

2691

Hierarchical Model:



2692
2693

Attributes:

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

2694
2695

r. Package

Element Name: Package

DTD Declaration:

```

    <!ELEMENT Package (Documentation*, (Package |
    BinaryCollaboration |
    MultiPartyCollaboration |
    BusinessTransaction)* ) >
    <!ATTLIST Package
    name CDATA #REQUIRED
    nameID ID #IMPLIED >
  
```

2703

Definition:

Defines a hierarchical name scope containing reusable elements.

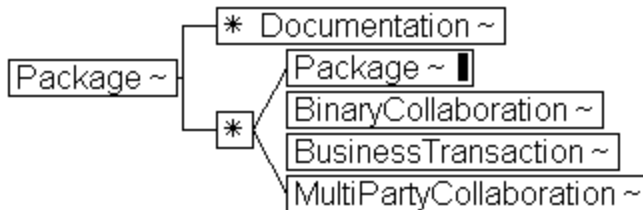
2707

Parents:

- ProcessSpecification
- Package

2709
2710
2711
2712

Hierarchical Model:



2713
2714
2715

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	Required Input

	model.	
nameID	XML ID version of name	Optional

2716
2717
2718

s. Performs

2719

Element Name: Performs

2720

DTD Declaration:

2721

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

2722

```
<!ATTLIST Performs
```

2723

```
    initiatingRole      CDATA      #REQUIRED
```

2724

```
    initiatingRoleIDRef IDREF      #IMPLIED
```

2725

```
    respondingRole     CDATA      #REQUIRED
```

2726

```
    respondingRoleIDRef IDREF      #IMPLIED >
```

2727

Definition:

2728

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

2729

2730

2731

Parents:

2732

- BusinessPartnerRole

2733

Attributes:

Attribute Name	Definition	Default Value
initiatingRole	The InitiatingRole that will be performed by the Business PartnerRole, qualified with the name of the BinaryCollaboration	Required Input
initiatingRoleIDRef	The XML IDREF version of InitiatingRole	Optional Input
respondingRole	The RespondingRole that will be performed by the Business PartnerRole, qualified with the name of the BinaryCollaboration	Required Input
respondingRoleIDRef	The XML IDREF version of RespondingRole	Optional Input

2734

2735

2736

2737

t. ProcessSpecification

2738

Element Name: ProcessSpecification

2739

DTD Declaration:

2740

```
<!ELEMENT ProcessSpecification (Documentation*,
SubstitutionSet*, (Include* | BusinessDocument* |
ProcessSpecification* | Package | BinaryCollaboration |
BusinessTransaction | MultiPartyCollaboration)*)>
```

2741

2742

2743

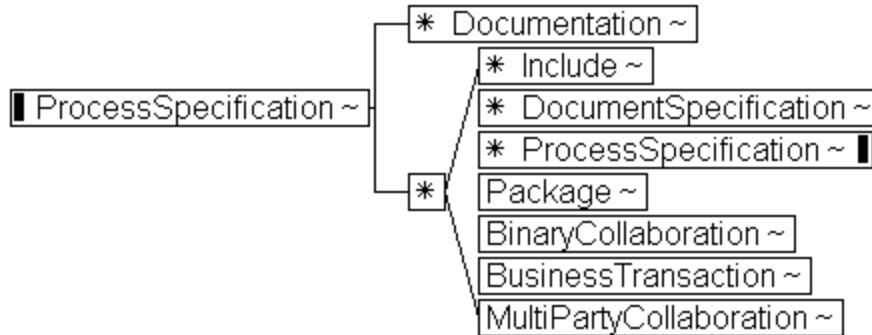
```

2744 <!ATTLIST ProcessSpecification
2745     name          ID          #REQUIRED
2746     version       CDATA       #REQUIRED
2747     uuid          CDATA       #REQUIRED >
2748
  
```

Definition:

2749 Root element of a process specification document that has a globally unique
 2750 identity.

2751 **Hierarchical Model:**



2752
 2753

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

2754

2755

2756 **u. Requesting Business Activity**

2757 **Element Name:** RequestingBusinessActivity

2758 **DTD Declaration:**

```

2759 <!ELEMENT RequestingBusinessActivity (Documentation*,
2760                                     DocumentEnvelope) >
2761 <!ATTLIST RequestingBusinessActivity
2762     name          CDATA          #IMPLIED
2763     nameID        ID             #IMPLIED
2764     isAuthorizationRequired (true | false) "false"
2765     isIntelligibleCheckRequired (true | false) "false"
2766     isNonRepudiationReceiptRequired (true | false) "false"
2767     isNonRepudiationRequired (true | false) "false"
  
```

2768	timeToAcknowledgeAcceptance	CDATA	#IMPLIED
2769	timeToAcknowledgeReceipt	CDATA	#IMPLIED>

2770

Definition:

2771

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.

2772

2773

2774

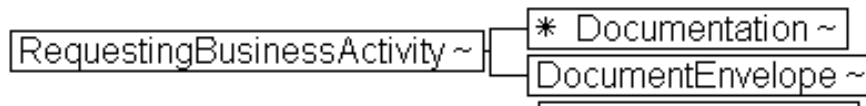
Parents:

2775

- BusinessTransaction

2776

2776

Hierarchical Model:

2777

2778

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.

2779

2780

2781

2782

v. Responding Business Activity

2783

Element Name: RespondingBusinessActivity

2784

DTD Declaration:

2785

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

2786

2787

```
<!ATTLIST RespondingBusinessActivity
```

2788

```
  name CDATA #IMPLIED
```

2789

```
  nameID ID #IMPLIED
```

2790

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

2791

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

2792

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

2793

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

2794

```
  timeToAcknowledgeReceipt CDATA #IMPLIED>
```

2795

Definition:

2796

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.

2797

2798

2799

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

2800

2801

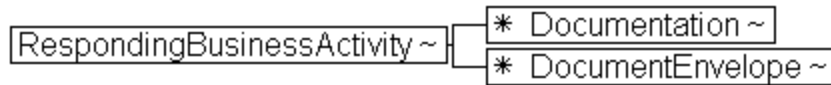
Parents:

2802

- BusinessTransaction

2803

Hierarchical Model:



2804
2805

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 value.

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

2806

2807

2808

2809

w. Responding Role

2810

XML Element Name: RespondingRole

2811

DTD Declaration:

2812

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

2813

```
<!ATTLIST RespondingRole
```

2814

```
  name CDATA #REQUIRED
```

2815

```
  nameID ID #IMPLIED>
```

2816

2817

Definition:

2818

A Responding Role is a role that is authorized to send the first response, e.g. the seller is authorized to send the acceptance of purchase order. This role is the responder in a binary collaboration.

2819

2820

2821

2822

Parents:

2823

BinaryCollaboration

2824

Attributes:

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

2825

x. Start

2826

Element Name: Start

2827

DTD Declaration:

2828

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

2829

```
<!ATTLIST Start
```

2830

```
  toBusinessState CDATA #REQUIRED
```

2831

```
  toBusinessStateIDRef IDREF #IMPLIED >
```

2832

Definition:

2833

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.

2834

2835

2836

Parents:

2837

- BinaryCollaboration

2838

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

2839

2840

2841

y. SubstitutionSet

2842

Element Name: SubstitutionSet

2843

DTD Declaration:

2844

```
<!ELEMENT SubstitutionSet (DocumentSubstitution |
AttributeSubstitution, Documentation)*>
```

2845

2846

```
<!ATTLIST SubstitutionSet
```

2847

```
  name CDATA #IMPLIED
```

2848

```
  nameID ID #IMPLIED
```

2849

```
  applyToScope CDATA #IMPLIED
```

2850

```
>
```

2851

Definition:

2852

A Substitution Set is a container for one or more AttributeSubstitution and/or DocumentSubstitution elements. The entire SubstitutionSet specifies document or attribute values that should be used in place of some documents and attribute values in an existing process specification.

2853

2854

2855

2856

Parents:

2857

- ProcessSpecification

2858

2859

Attributes:

2860

Attribute Name	Definition	Default Value
name	Name of the substitution set.	Required Input
nameID	The ID of the substitution set.	Required Input
applyToScope	Specifies the path to attributes or documents that are to be substituted for.	Required Input

2861

z. Success

2862

Element Name: Success

2863

DTD Declaration:

2864

```
<!ELEMENT Success (ConditionExpression, Documentation*)
```

2865

```
>
```

2866

```
<!ATTLIST Success
```

2867

```
  fromBusinessState CDATA #REQUIRED
```

2868

```
  conditionGuard (Success | BusinessFailure |
```


2869 TechnicalFailure | AnyFailure) #IMPLIED
 2870
 2871

Definition:

2872 Defines the successful conclusion of a binary collaboration as a transition from
 2873 an activity.

2874 **Parents:**

- BinaryCollaboration

2875 **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.
conditionGuard	The condition that guards this transition	Optional Valid Values: {Success, BusinessFailure, TechnicalFailure, AnyFailure}

2877

2878 aa. ConditionExpression

2879 **Element Name:** ConditionExpression

2880 **DTD Declaration:**

2881 <!ELEMENT ConditionExpression (Documentation*)>

2882 <!ATTLIST ConditionExpression

2883 expressionLanguage CDATA #IMPLIED

2884 expression CDATA #IMPLIED

2885 >**Definition:**

2886 Condition Expression is an expression that can be evaluated to TRUE or FALSE.

2887 **Parents:**

- Attachment

2889

2890

2891

Attributes:

Attribute Name	Definition	Default Value
expressionLanguage	The language of the expression, e.g. Java.	Required Input
expression	An expression whose evaluation results in TRUE or FALSE. For a transition, this determines whether this transition should happen or not. For a business document, this	Optional

	determines whether this is a valid business document for its envelope. The expression can refer to the name or content of the most recent DocumentEnvelope or content of documents within it.	
--	---	--

2892

bb. Transition

2893

ELEMENT Name: Transition

2894

DTD Declaration:

2895

```
<!ELEMENT Transition (ConditionExpression, Documentation*)
```

2896

```
>
```

2897

```
<!ATTLIST Transition
```

2898

```
  onInitiation (true | false) "false"
```

2899

```
  fromBusinessState CDATA #IMPLIED
```

2900

```
  fromBusinessStateIDRef IDREF #IMPLIED
```

2901

```
  toBusinessState CDATA #IMPLIED
```

2902

```
  toBusinessStateIDRef IDREF #IMPLIED
```

2903

```
  conditionGuard (Success | BusinessFailure |
```

2904

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

2905

Definition:

2906

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

2907

Choreography is expressed as transitions between business states.

2908

Parents:

2909

- BinaryCollaboration

2910

- BusinessPartnerRole

2911

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

conditionGuard	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}
-----------------------	---	--

2912
2913
2914
2915
2916
2917
2918

2919 **8.2 XML to UML cross-reference**

2920
2921
2922
2923

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
InitiatingRole	AuthorizedRole
RespondingRole	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
DocumentEnvelo	DocumentEnvelo

pe	pe
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

2924

2925

2926

2927

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:

2928

- BusinessState

2929

- CompletionState

2930

- BusinessActivity

2931

- BusinessAction

2932

- DocumentSecurity

2933

2934

2935

2936

2937 **8.3 Scoped Name Reference**

2938 The structure of ebXML process specifications encourages re-use. An
 2939 ebXMLProcessSpecification can include another ebXMLProcessSpecification by
 2940 reference.

2941 In addition the contents of a ebXMLProcessSpecification can be arranged in a
 2942 recursive package structure. The ebXMLProcessSpecification is a package
 2943 container, so it can contain packages within it. Package in itself is also a package
 2944 container, so it can contain further packages within it.

2945 Packages function as namespaces as per below.

2946 Finally a Package, at any level can have PackageContent. Types of
 2947 PackageContent are BusinessTransaction, BinaryCollaboration,
 2948 MultiPartyCollaboration.

2949 PackageContent are always uniquely named within a package. Lower level
 2950 elements a uniquely named within their parent PackageContent.

2951 Each PackageContent type is a built-in context provider for the core components
 2952 Logical Model for the Business Document definitions referenced by this
 2953 ebXMLProcessSpecification.

2954 Within a ebXMLProcessSpecification the following applies to naming:

2955 Specification elements reference other specification elements by name through
 2956 the use of attributes. The design pattern is that elements have a name attribute
 2957 and other elements that reference the named elements do so through an
 2958 attribute defined as the lowerCamelCase version of the referenced element (e.g.
 2959 AuthorizedRole has attribute name while Performs, which references
 2960 AuthorizedRole, has attribute authorizedRole). Two types of attributes are
 2961 provided for names and references, XML ID/IDREF based and plain text. Each
 2962 named element has a required name attribute and an optional nameID attribute.
 2963 Referencing elements have lowerCamelCase and lowerCamelCaseIDRef
 2964 attributes for the referenced element. XML ID/IDREF functionality requires all IDs
 2965 to be unique within a document and that all IDREFs point to a defined ID value.
 2966 Plain text attributes do not have this capability and may result in duplicate names.
 2967 To unambiguously identify a referenced element using plain text attribute in the
 2968 referencing attribute it is strongly recommended that XPath syntax be used.
 2969 However, this is not enforced in the DTD or Schema.

2970 The purpose of providing both solutions is to facilitate creation of Process
 2971 Specification Documents directly in XML and to support future development tools
 2972 that can automatically assign machine readable nameIDs and references. Both
 2973 styles can be used simultaneously, in which case the ID and IDREF versions
 2974 provide the unambiguous referencing and the plain text versions are used to
 2975 provide meaningful names. Examples of named elements and references:

```
2976 <Package name="ebXMLOrdering">
2977   <BinaryCollaboration name="OrderCollaboration" nameID="b112">
2978     <AuthorizedRole name="buyer" nameID="r224"/>
2979     <AuthorizedRole name="seller" nameID="r225"/>
2980   </BinaryCollaboration>
2981 </Package>
```

2982
 2983 <!--the XPath approach -->
 2984 <Performs
 2985 authorizedRole='//Package[@name="OAGOrdering"]/BinaryCollaboration[@name="OrderColla
 2986 boration"]/AuthorizedRole[@name="buyer"]'/>
 2987
 2988 <!--Combination approach -->
 2989 <Performs authorizedRole="buyer" authorizedRoleIDRef="r224"/>
 2990
 2991 It is not required to use the full path specification as shown above, other
 2992 forms of XPath expressions could be used as long as they resolve to a single
 2993 reference. For example if buyer was unique to the document then the XPath
 2994 could have been:
 2995 <Performs authorizedRole='//AuthorizedRole[@name="buyer"]'/>
 2996 Relative paths are also allowed for example:
 2997 <BusinessTransactionActivity fromAuthorizedRole='../AuthorizedRole[@name="buyer"]' ... />

2998 **8.4 Substitution Sets**

2999 There is a requirement for Business specifications that are less coupled to
 3000 technology and business details, such as specific document formats and
 3001 structures and timing parameters. Substitution sets support the capability to take
 3002 a generic business process and specialize it for a specific use. For example, an
 3003 ordering process may be very generic but a specific use of that process may
 3004 require specific document capabilities that go beyond the generic.

3005 A substitution set is placed in the more specific process specification and
 3006 replaces or makes more explicit document definition references and attribute
 3007 values.

3008 A Substitution Set is a container for one or more AttributeSubstitution and/or
 3009 DocumentSubstitution elements. The entire SubstitutionSet specifies document
 3010 or attribute values that should be used in place of some documents and attribute
 3011 values in an existing process specification.

3012

3013 **8.5 Sample XML document against above DTD**

3014
 3015 Provided in Appendix A
 3016

3017 **9 Business signal structures**

3018
 3019 The ebXML Message Service Specification signal structures provide
 3020 business service state alignment infrastructure, including unique message
 3021 identifiers and digests used to meet the basic process alignment

3022 requirements. The business signal payload structures provided herein
 3023 are optional and normative and are intended to provide business and
 3024 legal semantic to the business signals. Since signals do not differ in
 3025 structure from business transaction to business transaction, they are
 3026 defined once and for all, and their definition is implied by the conjunction
 3027 of the Business Process Specification Schema and Message Service
 3028 Specification. Here are the DTD's for business signal payload for
 3029 receiptAcknowledgment (from the RosettaNet website, courtesy of
 3030 RosettaNet, and Edifecs) and for acceptanceAcknowledgement and
 3031 exception.

3032 9.1.1 ReceiptAcknowledgment DTD

```

3033 <!--
3034
3035     RosettaNet XML Message Schema.
3036     http://www.rosettnet.org
3037     RosettaNet XML Message Schema.
3038     Receipt Acknowledgement
3039     Version 1.1
3040 -->
3041
3042 <!ENTITY % common-attributes "id CDATA #IMPLIED">
3043
3044 <!ELEMENT ReceiptAcknowledgement (
3045     fromRole ,
3046     NonRepudiationInformation? ,
3047     receivedDocumentDateTime ,
3048     receivedDocumentIdentifier ,
3049     thisMessageDateTime ,
3050     thisMessageIdentifier ,
3051     toRole ) >
3052
3053 <!ELEMENT fromRole
3054     ( PartnerRoleDescription ) >
3055
3056 <!ELEMENT PartnerRoleDescription (
3057     ContactInformation? ,
3058     GlobalPartnerRoleClassificationCode ,
3059     PartnerDescription ) >
3060
3061 <!ELEMENT ContactInformation (
3062     contactName ,
3063     EmailAddress ,
3064     telephoneNumber ) >
3065
3066 <!ELEMENT contactName
3067     ( FreeFormText ) >
3068
3069 <!ELEMENT FreeFormText
3070     ( #PCDATA ) >
3071
3072 <!ATTLIST FreeFormText
3073     xml:lang CDATA #IMPLIED >
  
```

```
3074
3075      <!ELEMENT EmailAddress
3076          ( #PCDATA ) >
3077
3078      <!ELEMENT telephoneNumber
3079          ( CommunicationsNumber ) >
3080
3081      <!ELEMENT CommunicationsNumber
3082          ( #PCDATA ) >
3083
3084      <!ELEMENT GlobalPartnerRoleClassificationCode
3085          ( #PCDATA ) >
3086
3087      <!ELEMENT PartnerDescription (
3088          BusinessDescription ,
3089          GlobalPartnerClassificationCode ) >
3090
3091      <!ELEMENT BusinessDescription (
3092          GlobalBusinessIdentifier ,
3093          GlobalSupplyChainCode ) >
3094
3095      <!ELEMENT GlobalBusinessIdentifier
3096          ( #PCDATA ) >
3097
3098      <!ELEMENT GlobalSupplyChainCode
3099          ( #PCDATA ) >
3100
3101      <!ELEMENT GlobalPartnerClassificationCode
3102          ( #PCDATA ) >
3103
3104      <!ELEMENT NonRepudiationInformation (
3105          GlobalDigestAlgorithmCode ,
3106          OriginalMessageDigest ) >
3107
3108      <!ELEMENT GlobalDigestAlgorithmCode
3109          ( #PCDATA ) >
3110
3111      <!ELEMENT OriginalMessageDigest
3112          ( #PCDATA ) >
3113
3114      <!ELEMENT receivedDocumentDateTime
3115          ( DateTimeStamp ) >
3116
3117      <!ELEMENT DateTimeStamp
3118          ( #PCDATA ) >
3119
3120      <!ELEMENT receivedDocumentIdentifier
3121          ( ProprietaryDocumentIdentifier ) >
3122
3123      <!ELEMENT ProprietaryDocumentIdentifier
3124          ( #PCDATA ) >
3125
3126      <!ELEMENT thisMessageDateTime
3127          ( DateTimeStamp ) >
3128
```



```

3129      <!ELEMENT thisMessageIdentifier
3130          ( ProprietaryMessageIdentifier ) >
3131
3132      <!ELEMENT ProprietaryMessageIdentifier
3133          ( #PCDATA ) >
3134
3135      <!ELEMENT toRole
3136          ( PartnerRoleDescription ) >
3137

```

9.1.2 AcceptanceAcknowledgement DTD

```

3138      <!--
3139          RosettaNet XML Message Schema.
3140          http://www.rosettnet.org
3141          RosettaNet XML Message Schema.
3142          Acceptance Acknowledgement Exception
3143          Version 1.1
3144      -->
3145
3146      <!ENTITY % common-attributes "id CDATA #IMPLIED">
3147
3148      <!ELEMENT AcceptanceAcknowledgementException (
3149          fromRole ,
3150          reason ,
3151          theMessageDatetime ,
3152          theOffendingDocumentDateTime ,
3153          theOffendingDocumentIdentifier ,
3154          thisMessageIdentifier ,
3155          toRole ) >
3156
3157      <!ELEMENT fromRole
3158          ( PartnerRoleDescription ) >
3159
3160      <!ELEMENT PartnerRoleDescription (
3161          ContactInformation? ,
3162          GlobalPartnerRoleClassificationCode ,
3163          PartnerDescription ) >
3164
3165      <!ELEMENT ContactInformation (
3166          contactName ,
3167          EmailAddress ,
3168          telephoneNumber ) >
3169
3170      <!ELEMENT contactName
3171          ( FreeFormText ) >
3172
3173      <!ELEMENT FreeFormText
3174          ( #PCDATA ) >
3175
3176      <!ATTLIST FreeFormText
3177          xml:lang CDATA #IMPLIED >
3178
3179      <!ELEMENT EmailAddress
3180          ( #PCDATA ) >
3181
3182      <!ELEMENT telephoneNumber

```

```
3183             ( CommunicationsNumber ) >
3184
3185     <!ELEMENT CommunicationsNumber
3186       ( #PCDATA ) >
3187
3188     <!ELEMENT GlobalPartnerRoleClassificationCode
3189       ( #PCDATA ) >
3190
3191     <!ELEMENT PartnerDescription (
3192       BusinessDescription ,
3193       GlobalPartnerClassificationCode ) >
3194
3195     <!ELEMENT BusinessDescription (
3196       GlobalBusinessIdentifier ,
3197       GlobalSupplyChainCode ) >
3198
3199     <!ELEMENT GlobalBusinessIdentifier
3200       ( #PCDATA ) >
3201
3202     <!ELEMENT GlobalSupplyChainCode
3203       ( #PCDATA ) >
3204
3205     <!ELEMENT GlobalPartnerClassificationCode
3206       ( #PCDATA ) >
3207
3208     <!ELEMENT reason
3209       ( FreeFormText ) >
3210
3211     <!ELEMENT theMessageDatetime
3212       ( DateTimeStamp ) >
3213
3214     <!ELEMENT DateTimeStamp
3215       ( #PCDATA ) >
3216
3217     <!ELEMENT theOffendingDocumentDateTime
3218       ( DateTimeStamp ) >
3219
3220     <!ELEMENT theOffendingDocumentIdentifier
3221       ( ProprietaryDocumentIdentifier ) >
3222
3223     <!ELEMENT ProprietaryDocumentIdentifier
3224       ( #PCDATA ) >
3225
3226     <!ELEMENT thisMessageIdentifier
3227       ( ProprietaryMessageIdentifier ) >
3228
3229     <!ELEMENT ProprietaryMessageIdentifier
3230       ( #PCDATA ) >
3231
3232     <!ELEMENT toRole
3233       ( PartnerRoleDescription ) >
3234
```

3235 9.1.3 Exception Signal DTD

3236

```
3237 <!--
3238     RosettaNet XML Message Schema.
3239     http://www.rosettanet.org
3240     RosettaNet XML Message Schema.
3241     Exception
3242     Version 1.1
3243 -->
3244
3245
3246 <!ENTITY % common-attributes "id CDATA #IMPLIED">
3247
3248 <!ELEMENT Exception (
3249     fromRole? ,
3250     reason ,
3251     theMessageDatetime ,
3252     theOffendingDocumentDateTime? ,
3253     theOffendingDocumentIdentifier? ,
3254     thisMessageIdentifier ,
3255     toRole? ) >
3256
3257 <!ELEMENT fromRole
3258     ( PartnerRoleDescription ) >
3259
3260 <!ELEMENT PartnerRoleDescription (
3261     ContactInformation? ,
3262     GlobalPartnerRoleClassificationCode? ,
3263     PartnerDescription? ) >
3264
3265 <!ELEMENT ContactInformation (
3266     contactName? ,
3267     EmailAddress? ,
3268     telephoneNumber? ) >
3269
3270 <!ELEMENT contactName
3271     ( FreeFormText ) >
3272
3273 <!ELEMENT FreeFormText
3274     ( #PCDATA ) >
3275 <!ATTLIST FreeFormText
3276     xml:lang CDATA #IMPLIED >
3277
3278 <!ELEMENT EmailAddress
3279     ( #PCDATA ) >
3280
3281 <!ELEMENT telephoneNumber
3282     ( CommunicationsNumber ) >
3283
3284 <!ELEMENT CommunicationsNumber
3285     ( #PCDATA ) >
3286
3287 <!ELEMENT GlobalPartnerRoleClassificationCode
3288     ( #PCDATA ) >
3289
3290 <!ELEMENT PartnerDescription (
3291     BusinessDescription? ,
```

```

3292         GlobalPartnerClassificationCode? ) >
3293
3294     <!ELEMENT BusinessDescription (
3295         GlobalBusinessIdentifier? ,
3296         GlobalSupplyChainCode? ) >
3297
3298     <!ELEMENT GlobalBusinessIdentifier
3299         ( #PCDATA ) >
3300
3301     <!ELEMENT GlobalSupplyChainCode
3302         ( #PCDATA ) >
3303
3304     <!ELEMENT GlobalPartnerClassificationCode
3305         ( #PCDATA ) >
3306
3307     <!ELEMENT reason
3308         ( FreeFormText ) >
3309
3310     <!ELEMENT theMessageDatetime
3311         ( DateTimeStamp ) >
3312
3313     <!ELEMENT DateTimeStamp
3314         ( #PCDATA ) >
3315
3316     <!ELEMENT theOffendingDocumentDateTime
3317         ( DateTimeStamp ) >
3318
3319     <!ELEMENT theOffendingDocumentIdentifier
3320         ( ProprietaryDocumentIdentifier ) >
3321
3322     <!ELEMENT ProprietaryDocumentIdentifier
3323         ( #PCDATA ) >
3324
3325     <!ELEMENT thisMessageIdentifier
3326         ( ProprietaryMessageIdentifier ) >
3327
3328     <!ELEMENT ProprietaryMessageIdentifier
3329         ( #PCDATA ) >
3330
3331     <!ELEMENT toRole
3332         ( PartnerRoleDescription ) >
3333
3334
3335

```

3336 10 Production Rules

3337 This section provides a set of production rules, defining the mapping from the
3338 UML version of the *Business Process Specification Schema* to the XML version.

3339 The primary purpose for these production rules is to govern the one-time
3340 generation of the DTD version of the *Business Process Specification Schema*
3341 from the UML Class Diagram version of *Business Process Specification Schema*.

- 3342 The Class Diagram version of *Business Process Specification Schema* is not
3343 intended for the direct creation of ebXML Business Process Specifications.
3344 However, if a *Business Process Specification* was in fact (programmatically)
3345 created as an instance of this class diagram, the production rules would also
3346 provide the prescriptive definition necessary to translate a such an instance into
3347 a XML Specification Document conformant with the DTD. The production rules
3348 are defined for concrete classes, abstract classes, aggregate associations,
3349 specialization associations and unidirectional associations.
- 3350 1. Classes are rendered as XML elements.
- 3351 2. Class attributes are rendered as XML attributes. NOTE: occurrence
3352 requirements (required vs optional) and default values for attributes are not
3353 modeled.
- 3354 3. Specialization classes (classes that inherit from another class) are rendered
3355 as XML elements including all attributes and aggregate associations from the
3356 base class. Repeated attributes are normalized to a single occurrence.
- 3357 4. Abstract classes are not rendered in the XML DTD. Abstract classes are
3358 inherited from and represent a form of collection. A class that aggregates an
3359 abstract class, essentially aggregates “any of each” of the specialization
3360 classes.
- 3361 5. An aggregate association renders the aggregated class as an XML child
3362 element with appropriate cardinality.
- 3363 6. A unidirectional association defines an attribute in the originating class of the
3364 same name as the class the association points to. This type of attribute is
3365 called a “reference attribute” and contains the name of the class it points to.
3366 The referenced class must have a “name” attribute.
- 3367 7. A class attribute data type, that has a class of the same name with stereotype
3368 <<Enumeration>> is rendered as an XML attribute enumeration. The
3369 Enumeration class does not have an explicit association.
- 3370 8. A class attribute data type (e.g. Time, URI, Boolean) that has no
3371 corresponding class definition is rendered as a string in the DTD. In the XML
3372 Schema version these data types are mapped as:
- 3373 Time - xsd:duration
3374 URI - xsd:anyURI
3375 Boolean - xsd:boolean
- 3376 9. Each class is given an optional “Documentation*” element which is intended
3377 for annotation of the specification instances. This is not modeled.
- 3378

3379 Appendix A: Sample XML Business Process 3380 Specification

```

3381
3382 <!-- edited by Kurt Kanaskie (Lucent Technologies) -->
3383 <!-- Notes from Kurt Kanaskie on 2001-04-17
3384 Differences from 099 DTD:
3385     binaryCollaboration attribute in Performs
3386     fromBinaryCollaboration attribute in Transition
3387     toBinaryCollaboration attribute in Transition
3388 Differences from original:
3389     guard instead of guardExpression
3390 See EBXMLSpecification_2001_04_23.dtd for other changes
3391 -->
3392 <!DOCTYPE ProcessSpecification SYSTEM "ebXMLProcessSpecification-
3393 v1.00.dtd">
3394 <ProcessSpecification name="Simple" version="1.1" uuid="[1234-5678-
3395 901234]">
3396     <!-- Business Documents -->
3397         <BusinessDocument name="Catalog Request"/>
3398         <BusinessDocument name="Catalog"/>
3399         <BusinessDocument name="Purchase Order"/>
3400         <BusinessDocument name="PO Acknowledgement"/>
3401         <BusinessDocument name="Credit Request"/>
3402         <BusinessDocument name="Credit Confirm"/>
3403         <BusinessDocument name="ASN"/>
3404         <BusinessDocument name="CreditAdvice"/>
3405         <BusinessDocument name="DebitAdvice"/>
3406         <BusinessDocument name="Invoice"/>
3407         <BusinessDocument name="Payment"/>
3408         <BusinessDocument name="Inventory Report Request"/>
3409         <BusinessDocument name="Inventory Report"/>
3410         <BusinessDocument name="Inventory Report"/>
3411     <Package name="Ordering">
3412         <!-- First the overall MultiParty Collaboration -->
3413         <MultiPartyCollaboration name="DropShip">
3414             <BusinessPartnerRole name="Customer">
3415                 <Performs initiatingRole="requestor"/>
3416                 <Performs initiatingRole="buyer"/>
3417                 <Transition fromBusinessState="Catalog Request"
3418 toBusinessState="Create Order"/>
3419             </BusinessPartnerRole>
3420             <BusinessPartnerRole name="Retailer">
3421                 <Performs respondingRole="provider"/>
3422                 <Performs respondingRole="seller"/>
3423                 <Performs initiatingRole="Creditor"/>
3424                 <Performs initiatingRole="buyer"/>
3425                 <Performs initiatingRole="Payee"/>
3426                 <Performs respondingRole="Payor"/>
3427                 <Performs initiatingRole="requestor"/>
3428                 <Transition fromBusinessState="Create Order"
3429 toBusinessState="Check Credit"/>

```

```

3430             <Transition fromBusinessState="Check Credit"
3431 toBusinessState="Create Order"/>
3432         </BusinessPartnerRole>
3433         <BusinessPartnerRole name="DropShip Vendor">
3434             <Performs respondingRole="seller"/>
3435             <Performs initiatingRole="payee"/>
3436             <Performs respondingRole="provider"/>
3437         </BusinessPartnerRole>
3438         <BusinessPartnerRole name="Credit Authority">
3439             <Performs respondingRole="credit service"/>
3440             <Performs respondingRole="payor"/>
3441         </BusinessPartnerRole>
3442     </MultiPartyCollaboration>
3443     <!-- Now the Binary Collaborations -->
3444     <BinaryCollaboration name="Request Catalog">
3445         <InitiatingRole name="requestor"/>
3446         <RespondingRole name="provider"/>
3447         <BusinessTransactionActivity name="Catalog Request"
3448 businessTransaction="Catalog Request" fromAuthorizedRole="requestor"
3449 toAuthorizedRole="provider"/>
3450     </BinaryCollaboration>
3451     <BinaryCollaboration name="Firm Order" timeToPerform="P2D">
3452         <Documentation>timeToPerform = Period: 2 days from
3453 start of transaction</Documentation>
3454         <InitiatingRole name="buyer"/>
3455         <RespondingRole name="seller"/>
3456         <BusinessTransactionActivity name="Create Order"
3457 businessTransaction="Create Order" fromAuthorizedRole="buyer"
3458 toAuthorizedRole="seller"/>
3459     </BinaryCollaboration>
3460     <BinaryCollaboration name="Product Fulfillment"
3461 timeToPerform="P5D">
3462         <Documentation>timeToPerform = Period: 5 days from
3463 start of transaction</Documentation>
3464         <InitiatingRole name="buyer"/>
3465         <RespondingRole name="seller"/>
3466         <BusinessTransactionActivity name="Create Order"
3467 businessTransaction="Create Order" fromAuthorizedRole="buyer"
3468 toAuthorizedRole="seller"/>
3469         <BusinessTransactionActivity name="Notify shipment"
3470 businessTransaction="Notify of advance shipment"
3471 fromAuthorizedRole="buyer" toAuthorizedRole="seller"/>
3472         <Start toBusinessState="Create Order"/>
3473         <Transition fromBusinessState="Create Order"
3474 toBusinessState="Notify shipment"/>
3475         <Success fromBusinessState="Notify shipment"
3476 conditionGuard="Success"/>
3477         <Failure fromBusinessState="Notify shipment"
3478 conditionGuard="BusinessFailure"/>
3479     </BinaryCollaboration>
3480     <BinaryCollaboration name="Inventory Status">
3481         <InitiatingRole name="requestor"/>
3482         <RespondingRole name="provider"/>

```

```
3483         <BusinessTransactionActivity name="Inventory Report
3484 Request" businessTransaction="Inventory Report Request"
3485 fromAuthorizedRole="requestor" toAuthorizedRole="provider"/>
3486         <BusinessTransactionActivity name="Inventory Report"
3487 businessTransaction="Inventory Report" fromAuthorizedRole="provider"
3488 toAuthorizedRole="requestor"/>
3489     </BinaryCollaboration>
3490     <BinaryCollaboration name="Credit Inquiry">
3491         <InitiatingRole name="creditor"/>
3492         <RespondingRole name="credit service"/>
3493         <BusinessTransactionActivity name="Check Credit"
3494 businessTransaction="Check Credit" fromAuthorizedRole="creditor"
3495 toAuthorizedRole="credit service"/>
3496     </BinaryCollaboration>
3497     <BinaryCollaboration name="Credit Payment">
3498         <InitiatingRole name="payee"/>
3499         <RespondingRole name="payor"/>
3500         <BusinessTransactionActivity name="Process Credit
3501 Payment" businessTransaction="Process Credit Payment"
3502 fromAuthorizedRole="payee" toAuthorizedRole="payor"/>
3503     </BinaryCollaboration>
3504     <!-- A compound BinaryCollaboration for illustration
3505 purposes-->
3506     <BinaryCollaboration name="Credit Charge">
3507         <InitiatingRole name="charger"/>
3508         <RespondingRole name="credit service"/>
3509         <CollaborationActivity name="Credit Inquiry"
3510 binaryCollaboration="Credit Inquiry" fromAuthorizedRole="charger"
3511 toAuthorizedRole="credit service"/>
3512         <CollaborationActivity name="Credit Payment"
3513 binaryCollaboration="Credit Payment" fromAuthorizedRole="charger"
3514 toAuthorizedRole="payor"/>
3515         <Transition fromBusinessState="Credit Inquiry"
3516 toBusinessState="Credit Payment"/>
3517     </BinaryCollaboration>
3518     <BinaryCollaboration name="Fulfillment Payment">
3519         <InitiatingRole name="payee"/>
3520         <RespondingRole name="payor"/>
3521         <BusinessTransactionActivity name="Process Payment"
3522 businessTransaction="Process Payment" fromAuthorizedRole="payee"
3523 toAuthorizedRole="payor"/>
3524     </BinaryCollaboration>
3525     <!-- Here are all the Business Transactions needed -->
3526     <BusinessTransaction name="Catalog Request">
3527         <RequestingBusinessActivity name="">
3528             <DocumentEnvelope isPositiveResponse="true"
3529 businessDocument="Catalog Request"/>
3530         </RequestingBusinessActivity>
3531         <RespondingBusinessActivity name="">
3532             <DocumentEnvelope isPositiveResponse="true"
3533 businessDocument="Catalog"/>
3534         </RespondingBusinessActivity>
3535     </BusinessTransaction>
3536     <BusinessTransaction name="Create Order">
```



```
3537         <RequestingBusinessActivity name=""
3538 isNonRepudiationRequired="true" timeToAcknowledgeReceipt="P2D"
3539 timeToAcknowledgeAcceptance="P3D">
3540         <DocumentEnvelope isPositiveResponse="true"
3541 businessDocument="Purchase Order"/>
3542     </RequestingBusinessActivity>
3543     <RespondingBusinessActivity name=""
3544 isNonRepudiationRequired="true" timeToAcknowledgeReceipt="P5D">
3545     <DocumentEnvelope isPositiveResponse="true"
3546 businessDocument="PO Acknowledgement"/>
3547     </RespondingBusinessActivity>
3548 </BusinessTransaction>
3549 <BusinessTransaction name="Check Credit ">
3550     <RequestingBusinessActivity name="">
3551     <DocumentEnvelope isPositiveResponse="true"
3552 businessDocument="Credit Request"/>
3553     </RequestingBusinessActivity>
3554     <RespondingBusinessActivity name="">
3555     <DocumentEnvelope isPositiveResponse="true"
3556 businessDocument="Credit Confirm"/>
3557     </RespondingBusinessActivity>
3558 </BusinessTransaction>
3559 <BusinessTransaction name="Notify of advance shipment">
3560     <RequestingBusinessActivity name="">
3561     <DocumentEnvelope isPositiveResponse="true"
3562 businessDocument="ASN"/>
3563     </RequestingBusinessActivity>
3564     <RespondingBusinessActivity name=""
3565 timeToAcknowledgeReceipt="P2D"/>
3566     </BusinessTransaction>
3567 <BusinessTransaction name="Process Credit Payment">
3568     <RequestingBusinessActivity name="">
3569     <DocumentEnvelope isPositiveResponse="true"
3570 businessDocument="CreditAdvice"/>
3571     </RequestingBusinessActivity>
3572     <RespondingBusinessActivity name="">
3573     <DocumentEnvelope isPositiveResponse="true"
3574 businessDocument="DebitAdvice"/>
3575     </RespondingBusinessActivity>
3576 </BusinessTransaction>
3577 <BusinessTransaction name="Process Payment">
3578     <RequestingBusinessActivity name="">
3579     <DocumentEnvelope isPositiveResponse="true"
3580 businessDocument="Invoice"/>
3581     </RequestingBusinessActivity>
3582     <RespondingBusinessActivity name="">
3583     <DocumentEnvelope isPositiveResponse="true"
3584 businessDocument="Payment"/>
3585     </RespondingBusinessActivity>
3586 </BusinessTransaction>
3587 <BusinessTransaction name="Request Inventory Report">
3588     <RequestingBusinessActivity name="">
3589     <DocumentEnvelope isPositiveResponse="true"
3590 businessDocument="Inventory Report Request"/>
3591     </RequestingBusinessActivity>
```

```
3592         <RespondingBusinessActivity name="">
3593             <DocumentEnvelope isPositiveResponse="true"
3594 businessDocument="Inventory Report"/>
3595         </RespondingBusinessActivity>
3596     </BusinessTransaction>
3597     <BusinessTransaction name="Inventory Report">
3598         <RequestingBusinessActivity name="">
3599             <DocumentEnvelope isPositiveResponse="true"
3600 businessDocument="Inventory Report"/>
3601         </RequestingBusinessActivity>
3602         <RespondingBusinessActivity name=""/>
3603     </BusinessTransaction>
3604 </Package>
3605 </ProcessSpecification>
```

3606 **Appendix B: Business Process Specification Schema**

3607 **DTD**

```

3608
3609 <!-- ===== --
3610 >
3611 <!-- Editor: Kurt Kanaskie (Lucent Technologies) --
3612 >
3613 <!-- Version: Version 1.0 --
3614 >
3615 <!-- Updated: 2001-05-10 --
3616 >
3617 <!-- --
3618 >
3619 <!-- Public Identifier: --
3620 >
3621 <!-- "-//ebXML//DTD Process Specification ver 1.0//EN" --
3622 >
3623 <!-- --
3624 >
3625 <!-- Purpose: --
3626 >
3627 <!-- The ebXML Specification DTD provides a standard --
3628 >
3629 <!-- framework by which business systems may be --
3630 >
3631 <!-- configured to support execution of business --
3632 >
3633 <!-- transactions. It is based upon prior UN/CEFACT --
3634 >
3635 <!-- work, specifically the metamodel behind the --
3636 >
3637 <!-- UN/CEFACT Unified Modeling Methodology (UMM) defined --
3638 >
3639 <!-- in the N090R9.1 specification. --
3640 -->
3641 <!-- --
3642 >
3643 <!-- The Specification Schema supports the specification --
3644 >
3645 <!-- of Business Transactions and the choreography of --
3646 >
3647 <!-- Business Transactions into Business Collaborations. --
3648 >
3649 <!-- --
3650 >
3651 <!-- Notes: --
3652 >
3653 <!-- time periods are represented using ISO 8601 format --
3654 >
3655 <!-- (e.g. P2D for 2 Days, P2H30M for 2 Hours 30 Minutes --
3656 >

```

```

3657 <!-- --
3658 >
3659 <!-- Naming and reference is based on convention that an --
3660 >
3661 <!-- Element with a name attribute (e.g. AuthorizedRole) --
3662 >
3663 <!-- is refernced by an attribute in another element with --
3664 >
3665 <!-- the name in lowerCamelCase (e.g. authorizedRole). --
3666 >
3667 <!-- --
3668 >
3669 <!-- fromBusinessState and toBusinessState refer to the --
3670 >
3671 <!-- the names of a BusinessTransactionActivity, --
3672 >
3673 <!-- CollaborationActivity, Fork, and Join, all are targets for --
3674 >
3675 <!-- from/to in Transition. This deviates from the normal --
3676 >
3677 <!-- convention of lowerCamelCase attribute name --
3678 >
3679 <!-- BusinessState is used as a generic term for: --
3680 >
3681 <!-- Fork, Join, Success, Failure --
3682 >
3683 <!-- --
3684 >
3685 <!-- Constraints: --
3686 >
3687 <!-- - attributes location, logicalModel, pattern, specification --
3688 >
3689 <!-- uri, are of type xsd:anyURI -->
3690 <!-- - attributes timeTo* are of type xsd:duration --
3691 >
3692 <!-- isSuccess is an expression (e.g. XPath) that results --
3693 >
3694 <!-- in a boolean true or false based on document name or --
3695 >
3696 <!-- document content. --
3697 >
3698 <!-- --
3699 >
3700 <!-- ===== --
3701 >
3702
3703 <!ELEMENT ProcessSpecification (Documentation*, SubstitutionSet*,
3704 (Include* | BusinessDocument* | ProcessSpecification* |
3705 Package | BinaryCollaboration |
3706 BusinessTransaction | MultiPartyCollaboration)*)>
3707 <!ATTLIST ProcessSpecification
3708 name ID #REQUIRED
3709 uuid CDATA #REQUIRED
3710 version CDATA #REQUIRED
3711 >

```

```
3712 <!ELEMENT Documentation (#PCDATA)>
3713 <!ATTLIST Documentation
3714     uri CDATA #IMPLIED
3715 >
3716
3717 <!ELEMENT Include (Documentation*)>
3718 <!ATTLIST Include
3719     name CDATA #REQUIRED
3720     uuid CDATA #REQUIRED
3721     uri CDATA #REQUIRED
3722     version CDATA #REQUIRED
3723 >
3724
3725 <!ELEMENT BusinessDocument (ConditionExpression?, Documentation*)>
3726 <!ATTLIST BusinessDocument
3727     name CDATA #REQUIRED
3728     nameID ID #IMPLIED
3729     specificationLocation CDATA #IMPLIED
3730     specificationElement CDATA #IMPLIED
3731 >
3732
3733 <!ELEMENT ConditionExpression (Documentation*)>
3734 <!ATTLIST ConditionExpression
3735     expressionLanguage CDATA #IMPLIED
3736     expression CDATA #IMPLIED
3737 >
3738
3739 <!ELEMENT SubstitutionSet (DocumentSubstitution | AttributeSubstitution
3740 | Documentation)*>
3741 <!ATTLIST SubstitutionSet
3742     name CDATA #IMPLIED
3743     nameId IDREF #IMPLIED
3744     applyToScope CDATA #IMPLIED
3745 >
3746
3747 <!ELEMENT DocumentSubstitution (Documentation*)>
3748 <!ATTLIST DocumentSubstitution
3749     originalBusinessDocument CDATA #IMPLIED
3750     originalBusinessDocumentID IDREF #IMPLIED
3751     substituteBusinessDocument CDATA #IMPLIED
3752     substituteBusinessDocumentID IDREF #IMPLIED
3753 >
3754
3755 <!ELEMENT AttributeSubstitution (Documentation*)>
3756 <!ATTLIST AttributeSubstitution
3757     attributeName CDATA #IMPLIED
3758     value CDATA #IMPLIED
3759 >
3760
3761 <!ELEMENT Package (Documentation*,
3762     (Package | BinaryCollaboration | BusinessTransaction |
3763 MultiPartyCollaboration)*)>
3764 <!ATTLIST Package
3765     name CDATA #REQUIRED
3766     nameID ID #IMPLIED
```

```

3767 >
3768
3769 <!ELEMENT BinaryCollaboration (Documentation*, InitiatingRole,
3770 RespondingRole,
3771 (Documentation* | Start | Transition | Success
3772 | Failure |
3773 BusinessTransactionActivity |
3774 CollaborationActivity | Fork | Join)*>
3775 <!ATTLIST BinaryCollaboration
3776 name CDATA #REQUIRED
3777 nameID ID #IMPLIED
3778 pattern CDATA #IMPLIED
3779 beginsWhen CDATA #IMPLIED
3780 endsWhen CDATA #IMPLIED
3781 preCondition CDATA #IMPLIED
3782 postCondition CDATA #IMPLIED
3783 timeToPerform CDATA #IMPLIED
3784 >
3785 <!ELEMENT MultiPartyCollaboration (Documentation*,
3786 BusinessPartnerRole*)>
3787 <!ATTLIST MultiPartyCollaboration
3788 name CDATA #REQUIRED
3789 nameID ID #IMPLIED
3790 >
3791
3792 <!ELEMENT InitiatingRole (Documentation*)>
3793 <!ATTLIST InitiatingRole
3794 name CDATA #REQUIRED
3795 nameID ID #IMPLIED
3796 >
3797 <!ELEMENT RespondingRole(Documentation*)>
3798 <!ATTLIST RespondingRole
3799 name CDATA #REQUIRED
3800 nameID ID #IMPLIED
3801 >
3802
3803 <!-- A BusinessState is one of Start, Success, Failure, Fork, Join,
3804 BusinessTransactionActivity or CollaborationActivity -->
3805 <!-- fromBusinessState and toBusinessState are fully qualified using
3806 XPath -->
3807 <!ELEMENT Transition (ConditionExpression?, Documentation*)>
3808 <!ATTLIST Transition
3809 onInitiation (true | false) "false"
3810 fromBusinessState CDATA #IMPLIED
3811 fromBusinessStateIDRef IDREF #IMPLIED
3812 toBusinessState CDATA #IMPLIED
3813 toBusinessStateIDRef IDREF #IMPLIED
3814 conditionGuard (Success | BusinessFailure | TechnicalFailure |
3815 AnyFailure ) #IMPLIED
3816 >
3817 <!-- Start is a special type of Transition in that it only has a
3818 destination -->
3819 <!ELEMENT Start (Documentation*)>
3820 <!ATTLIST Start
3821 toBusinessState CDATA #REQUIRED

```

```
3822         toBusinessStateIDRef IDREF #IMPLIED
3823     >
3824     <!-- Success is a special type of Transition in that it only has a
3825     origination -->
3826     <!ELEMENT Success (ConditionExpression?, Documentation*)>
3827     <!ATTLIST Success
3828         fromBusinessState CDATA #REQUIRED
3829         fromBusinessStateIDRef IDREF #IMPLIED
3830         conditionGuard (Success | BusinessFailure | TechnicalFailure |
3831 AnyFailure ) #IMPLIED
3832     >
3833     <!-- Failure is a special type of Transition in that it only has a
3834     origination -->
3835     <!ELEMENT Failure (ConditionExpression?, Documentation*)>
3836     <!ATTLIST Failure
3837         fromBusinessState CDATA #REQUIRED
3838         fromBusinessStateIDRef IDREF #IMPLIED
3839         conditionGuard (Success | BusinessFailure | TechnicalFailure |
3840 AnyFailure ) #IMPLIED
3841     >
3842
3843     <!-- Fork is a special type of BusinessState that can be transitioned
3844     to -->
3845     <!ELEMENT Fork (Documentation*)>
3846     <!ATTLIST Fork
3847         name CDATA #REQUIRED
3848         nameID ID #IMPLIED
3849     >
3850     <!-- Join is a special type of BusinessState that can be transitioned
3851     to -->
3852     <!ELEMENT Join (Documentation*)>
3853     <!ATTLIST Join
3854         name CDATA #REQUIRED
3855         nameID ID #IMPLIED
3856         waitForAll (true | false) "true"
3857     >
3858
3859     <!-- fromAuthorizedRole and toAuthorizedRole are fully qualified using
3860     XPath -->
3861     <!-- BusinessTransactionActivity is a BusinessState that can be
3862     transitioned to -->
3863     <!ELEMENT BusinessTransactionActivity (Documentation*)>
3864     <!ATTLIST BusinessTransactionActivity
3865         name CDATA #REQUIRED
3866         nameID ID #IMPLIED
3867         businessTransaction CDATA #REQUIRED
3868         businessTransactionIDRef IDREF #IMPLIED
3869         fromAuthorizedRole CDATA #REQUIRED
3870         fromAuthorizedRoleIDRef IDREF #IMPLIED
3871         toAuthorizedRole CDATA #REQUIRED
3872         toAuthorizedRoleIDRef IDREF #IMPLIED
3873         isConcurrent (true | false) "true"
3874         isLegallyBinding (true | false) "true"
3875         timeToPerform CDATA #IMPLIED
3876     >
```

```
3877
3878 <!-- fromAuthorizedRole and toAuthorizedRole are fully qualified using
3879 XPath -->
3880 <!-- CollaborationActivity is a BusinessState that can be transitioned
3881 to -->
3882 <!ELEMENT CollaborationActivity (Documentation*)>
3883 <!ATTLIST CollaborationActivity
3884     name CDATA #REQUIRED
3885     nameID ID #IMPLIED
3886     fromAuthorizedRole CDATA #REQUIRED
3887     fromAuthorizedRoleIDRef IDREF #IMPLIED
3888     toAuthorizedRole CDATA #REQUIRED
3889     toAuthorizedRoleIDRef IDREF #IMPLIED
3890     binaryCollaboration CDATA #REQUIRED
3891     binaryCollaborationIDRef IDREF #IMPLIED
3892 >
3893 <!ELEMENT BusinessTransaction (Documentation*,
3894 RequestingBusinessActivity, RespondingBusinessActivity)>
3895 <!ATTLIST BusinessTransaction
3896     name CDATA #REQUIRED
3897     nameID ID #IMPLIED
3898     pattern CDATA #IMPLIED
3899     beginsWhen CDATA #IMPLIED
3900     endsWhen CDATA #IMPLIED
3901     isGuaranteedDeliveryRequired (true | false) "false"
3902     preCondition CDATA #IMPLIED
3903     postCondition CDATA #IMPLIED
3904 >
3905 <!ELEMENT RequestingBusinessActivity (Documentation*,
3906 DocumentEnvelope)>
3907 <!ATTLIST RequestingBusinessActivity
3908     name CDATA #IMPLIED
3909     nameID ID #IMPLIED
3910     isAuthorizationRequired (true | false) "false"
3911     isIntelligibleCheckRequired (true | false) "false"
3912     isNonRepudiationReceiptRequired (true | false) "false"
3913     isNonRepudiationRequired (true | false) "false"
3914     timeToAcknowledgeAcceptance CDATA #IMPLIED
3915     timeToAcknowledgeReceipt CDATA #IMPLIED
3916 >
3917 <!ELEMENT RespondingBusinessActivity (Documentation*,
3918 DocumentEnvelope*)>
3919 <!ATTLIST RespondingBusinessActivity
3920     name CDATA #IMPLIED
3921     nameID ID #IMPLIED
3922     isAuthorizationRequired (true | false) "false"
3923     isIntelligibleCheckRequired (true | false) "false"
3924     isNonRepudiationReceiptRequired (true | false) "false"
3925     isNonRepudiationRequired (true | false) "false"
3926     timeToAcknowledgeReceipt CDATA #IMPLIED
3927 >
3928
3929 <!ELEMENT DocumentEnvelope (Documentation*, Attachment*)>
3930 <!ATTLIST DocumentEnvelope
3931     businessDocument CDATA #REQUIRED
```



```
3932     businessDocumentIDRef IDREF #IMPLIED
3933     isPositiveResponse (true | false) "false"
3934     isAuthenticated (true | false) "false"
3935     isConfidential (true | false) "false"
3936     isTamperProof (true | false) "false"
3937 >
3938 <!ELEMENT Attachment (Documentation*)>
3939 <!ATTLIST Attachment
3940     name CDATA #REQUIRED
3941     nameID ID #IMPLIED
3942     businessDocument CDATA #IMPLIED
3943     businessDocumentIDRef IDREF #IMPLIED
3944     mimeType CDATA #REQUIRED
3945     specification CDATA #IMPLIED
3946     version CDATA #IMPLIED
3947     isAuthenticated (true | false) "false"
3948     isConfidential (true | false) "false"
3949     isTamperProof (true | false) "false"
3950 >
3951 <!ELEMENT BusinessPartnerRole (Documentation*, Performs*, Transition*)>
3952 <!ATTLIST BusinessPartnerRole
3953     name CDATA #REQUIRED
3954     nameID ID #IMPLIED
3955 >
3956
3957 <!-- authorizedRole is fully qualified using XPath -->
3958 <!ELEMENT Performs (Documentation*)>
3959 <!ATTLIST Performs
3960     initiatingRole CDATA #IMPLIED
3961     initiatingRoleIDRef IDREF #IMPLIED
3962     respondingRole CDATA #IMPLIED
3963     respondingRoleIDRef IDREF #IMPLIED
3964 >
3965 >
```

3966 **Appendix C: Business Process Specification Schema**

3967 **XML Schema**

```
3968 <?xml version="1.0" encoding="UTF-8"?>
3969 <!-- edited by Kurt Kanaskie (Lucent Technologies) -->
3970 <!-- Updated 2001-05-10
3971 <!-- Kanaskie Updated 2001-04-27
3972         Use uriReference instead of anyURI
3973         Use timeDuration instead of duration
3974 -->
3975 <!-- Kanaskie Changed 2001-04-27
3976     See DTD for list of changes.
3977     Differences from DTD version:
3978     AuthorizedRole minOccurs=2 maxOccurs=2
3979     <xsd:attribute name="pattern" type="xsd:anyURI"/>
3980     <xsd:attribute name="uri" type="xsd:anyURI" use="required"/>
3981     <xsd:attribute name="location" type="xsd:anyURI"/>
3982     <xsd:attribute name="logicalModel" type="xsd:anyURI"/>
3983     <xsd:attribute name="specification" type="xsd:anyURI"/>
3984     <xsd:attribute name="timeToPerform" type="xsd:duration"/>
3985     <xsd:attribute name="timeToPerform" type="xsd:duration"/>
3986     <xsd:attribute name="timeToAcknowledgeAcceptance"
3987 type="xsd:duration"/>
3988     <xsd:attribute name="timeToAcknowledgeReceipt"
3989 type="xsd:duration"/>
3990     <xsd:attribute name="timeToAcknowledgeAcceptance"
3991 type="xsd:duration"/>
3992     <xsd:attribute name="timeToAcknowledgeReceipt"
3993 type="xsd:duration"/>
3994     <xsd:attribute name="isAuthenticated" type="xsd:boolean"
3995 value="false"/>
3996     <xsd:attribute name="isConfidential" type="xsd:boolean"
3997 value="false"/>
3998     <xsd:attribute name="isTamperProof" type="xsd:boolean"
3999 value="false"/>
4000     <xsd:attribute name="isGuaranteedDeliveryRequired"
4001 type="xsd:boolean" value="false"/>
4002     <xsd:attribute name="isConcurrent" type="xsd:boolean"
4003 value="true"/>
4004     <xsd:attribute name="isLegallyBinding" type="xsd:boolean"
4005 value="true"/>
4006     <xsd:attribute name="isAuthenticated" type="xsd:boolean"
4007 value="false"/>
4008     <xsd:attribute name="isConfidential" type="xsd:boolean"
4009 value="false"/>
4010     <xsd:attribute name="isTamperProof" type="xsd:boolean"
4011 value="false"/>
4012     <xsd:attribute name="waitForAll" type="xsd:boolean"
4013 value="true"/>
4014     <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean"
4015 value="false"/>
4016     <xsd:attribute name="isIntelligibleCheckRequired"
4017 type="xsd:boolean" value="false"/>
```

```
4018         <xsd:attribute name="isNonRepudiationReceiptRequired"
4019 type="xsd:boolean" value="false"/>
4020         <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean"
4021 value="false"/>
4022         <xsd:attribute name="isAuthorizationRequired" type="xsd:boolean"
4023 value="false"/>
4024         <xsd:attribute name="isIntelligibleCheckRequired"
4025 type="xsd:boolean" value="false"/>
4026         <xsd:attribute name="isNonRepudiationReceiptRequired"
4027 type="xsd:boolean" value="false"/>
4028         <xsd:attribute name="isNonRepudiationRequired" type="xsd:boolean"
4029 value="false"/>
4030         <xsd:attribute name="onInitiation" type="xsd:boolean"
4031 value="false"/>
4032     -->
4033 <xsd:schema targetNamespace="http://www.ebxml.org/BusinessProcess"
4034 xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
4035 xmlns="http://www.ebxml.org/BusinessProcess"
4036 elementFormDefault="qualified">
4037     <xsd:element name="Attachment">
4038         <xsd:complexType>
4039             <xsd:sequence>
4040                 <xsd:element ref="Documentation" minOccurs="0"
4041 maxOccurs="unbounded"/>
4042             </xsd:sequence>
4043             <xsd:attribute name="name" type="xsd:string"
4044 use="required"/>
4045             <xsd:attribute name="nameID" type="xsd:ID"/>
4046             <xsd:attribute name="businessDocument"
4047 type="xsd:string"/>
4048             <xsd:attribute name="businessDocumentIDRef"
4049 type="xsd:IDREF"/>
4050             <xsd:attribute name="specification"
4051 type="xsd:uriReference"/>
4052             <xsd:attribute name="mimeType" type="xsd:string"
4053 use="required"/>
4054             <xsd:attribute name="version" type="xsd:string"/>
4055             <xsd:attribute name="isAuthenticated"
4056 type="xsd:boolean" value="false"/>
4057             <xsd:attribute name="isConfidential"
4058 type="xsd:boolean" value="false"/>
4059             <xsd:attribute name="isTamperProof"
4060 type="xsd:boolean" value="false"/>
4061         </xsd:complexType>
4062     </xsd:element>
4063     <xsd:element name="InitiatingRole">
4064         <xsd:complexType>
4065             <xsd:sequence>
4066                 <xsd:element ref="Documentation" minOccurs="0"
4067 maxOccurs="unbounded"/>
4068             </xsd:sequence>
4069             <xsd:attribute name="name" type="xsd:string"
4070 use="required"/>
4071             <xsd:attribute name="nameID" type="xsd:ID"/>
4072         </xsd:complexType>
```

```
4073         </xsd:element>
4074         <xsd:element name="RespondingRole">
4075             <xsd:complexType>
4076                 <xsd:sequence>
4077                     <xsd:element ref="Documentation" minOccurs="0"
4078 maxOccurs="unbounded"/>
4079                 </xsd:sequence>
4080                 <xsd:attribute name="name" type="xsd:string"
4081 use="required"/>
4082                 <xsd:attribute name="nameID" type="xsd:ID"/>
4083             </xsd:complexType>
4084         </xsd:element>
4085         <xsd:element name="BinaryCollaboration">
4086             <xsd:complexType>
4087                 <xsd:sequence>
4088                     <xsd:element ref="Documentation" minOccurs="0"
4089 maxOccurs="unbounded"/>
4090                     <xsd:element ref="InitiatingRole"/>
4091                     <xsd:element ref="RespondingRole"/>
4092                     <xsd:choice minOccurs="0"
4093 maxOccurs="unbounded">
4094                         <xsd:element ref="Documentation"
4095 minOccurs="0" maxOccurs="unbounded"/>
4096                         <xsd:element ref="Start"/>
4097                         <xsd:element ref="Transition"/>
4098                         <xsd:element ref="Success"/>
4099                         <xsd:element ref="Failure"/>
4100                         <xsd:element
4101 ref="BusinessTransactionActivity"/>
4102                         <xsd:element
4103 ref="CollaborationActivity"/>
4104                         <xsd:element ref="Fork"/>
4105                         <xsd:element ref="Join"/>
4106                     </xsd:choice>
4107                 </xsd:sequence>
4108                 <xsd:attribute name="name" type="xsd:string"
4109 use="required"/>
4110                 <xsd:attribute name="nameID" type="xsd:ID"/>
4111                 <xsd:attribute name="pattern"
4112 type="xsd:uriReference"/>
4113                 <xsd:attribute name="beginsWhen" type="xsd:string"/>
4114                 <xsd:attribute name="endsWhen" type="xsd:string"/>
4115                 <xsd:attribute name="preCondition"
4116 type="xsd:string"/>
4117                 <xsd:attribute name="postCondition"
4118 type="xsd:string"/>
4119                 <xsd:attribute name="timeToPerform"
4120 type="xsd:timeDuration"/>
4121             </xsd:complexType>
4122         </xsd:element>
4123         <xsd:element name="BusinessDocument">
4124             <xsd:complexType>
4125                 <xsd:sequence >
4126                     <xsd:element ref="Documentation" minOccurs="0"
4127 maxOccurs="unbounded"/>
```

```
4128             <xsd:element ref="ConditionExpression"
4129 minOccurs="0" maxOccurs="1"/>
4130         </xsd:sequence>
4131     <xsd:attribute name="name" type="xsd:string"
4132 use="required"/>
4133     <xsd:attribute name="nameID" type="xsd:ID"/>
4134     <xsd:attribute name="specificationLocation"
4135 type="xsd:string"/>
4136     <xsd:attribute name="specificationElement"
4137 type="xsd:string"/>
4138 </xsd:complexType>
4139 </xsd:element>
4140 <xsd:element name="SubstitutionSet">
4141     <xsd:complexType>
4142         <xsd:sequence>
4143             <xsd:element ref="DocumentSubstitution"
4144 minOccurs="0" maxOccurs="unbounded"/>
4145             <xsd:element ref="AttributeSubstitution"
4146 minOccurs="0" maxOccurs="unbounded"/>
4147             <xsd:element ref="Documentation" minOccurs="0"
4148 maxOccurs="unbounded"/>
4149         </xsd:sequence>
4150         <xsd:attribute name="name" type="xsd:string"/>
4151         <xsd:attribute name="nameId" type="xsd:ID"/>
4152         <xsd:attribute name="applyToScope"
4153 type="xsd:string"/>
4154     </xsd:complexType>
4155 </xsd:element>
4156 <xsd:element name="DocumentSubstitution">
4157     <xsd:complexType>
4158         <xsd:element ref="Documentation" minOccurs="0"
4159 maxOccurs="unbounded"/>
4160         <xsd:attribute name="originalBusinessDocument"
4161 type="xsd:string"/>
4162         <xsd:attribute name="originalBusinessDocumentID"
4163 type="xsd:ID"/>
4164         <xsd:attribute name="substituteBusinessDocument"
4165 type="xsd:string"/>
4166         <xsd:attribute name="substituteBusinessDocumentId"
4167 type="xsd:ID"/>
4168     </xsd:complexType>
4169 </xsd:element>
4170 <xsd:element name="AttributeSubstitution">
4171     <xsd:complexType>
4172         <xsd:sequence>
4173             <xsd:element ref="Documentation" minOccurs="0"
4174 maxOccurs="unbounded"/>
4175         </xsd:sequence>
4176         <xsd:attribute name="attributeName"
4177 type="xsd:string"/>
4178         <xsd:attribute name="value" type="xsd:string"/>
4179     </xsd:complexType>
4180 </xsd:element>
4181 <xsd:element name="ConditionExpression">
4182     <xsd:complexType>
```

```
4183         <xsd:sequence>
4184             <xsd:element ref="Documentation" minOccurs="0"
4185 maxOccurs="unbounded" />
4186         </xsd:sequence>
4187         <xsd:attribute name="expressionLanguage"
4188 type="xsd:string" />
4189         <xsd:attribute name="expression" type="xsd:string" />
4190     </xsd:complexType>
4191 </xsd:element>
4192
4193
4194     <xsd:element name="BusinessPartnerRole">
4195         <xsd:complexType>
4196             <xsd:sequence>
4197                 <xsd:element ref="Documentation" minOccurs="0"
4198 maxOccurs="unbounded" />
4199                 <xsd:element ref="Performs" minOccurs="0"
4200 maxOccurs="unbounded" />
4201                 <xsd:element ref="Transition" minOccurs="0"
4202 maxOccurs="unbounded" />
4203             </xsd:sequence>
4204             <xsd:attribute name="name" type="xsd:string"
4205 use="required" />
4206             <xsd:attribute name="nameID" type="xsd:ID" />
4207         </xsd:complexType>
4208     </xsd:element>
4209     <xsd:element name="BusinessTransaction">
4210         <xsd:complexType>
4211             <xsd:sequence>
4212                 <xsd:element ref="Documentation" minOccurs="0"
4213 maxOccurs="unbounded" />
4214                 <xsd:element ref="RequestingBusinessActivity" />
4215                 <xsd:element ref="RespondingBusinessActivity" />
4216             </xsd:sequence>
4217             <xsd:attribute name="name" type="xsd:string"
4218 use="required" />
4219             <xsd:attribute name="nameID" type="xsd:ID" />
4220             <xsd:attribute name="pattern"
4221 type="xsd:uriReference" />
4222             <xsd:attribute name="beginsWhen" type="xsd:string" />
4223             <xsd:attribute name="endsWhen" type="xsd:string" />
4224             <xsd:attribute name="isGuaranteedDeliveryRequired"
4225 type="xsd:boolean" value="false" />
4226             <xsd:attribute name="preCondition"
4227 type="xsd:string" />
4228             <xsd:attribute name="postCondition"
4229 type="xsd:string" />
4230         </xsd:complexType>
4231     </xsd:element>
4232     <xsd:element name="BusinessTransactionActivity">
4233         <xsd:complexType>
4234             <xsd:sequence>
4235                 <xsd:element ref="Documentation" minOccurs="0"
4236 maxOccurs="unbounded" />
4237             </xsd:sequence>
```

```

4238         <xsd:attribute name="name" type="xsd:string"
4239 use="required" />
4240         <xsd:attribute name="nameID" type="xsd:ID" />
4241         <xsd:attribute name="businessTransaction"
4242 type="xsd:string" use="required" />
4243         <xsd:attribute name="businessTransactionIDRef"
4244 type="xsd:IDREF" />
4245         <xsd:attribute name="fromAuthorizedRole"
4246 type="xsd:string" use="required" />
4247         <xsd:attribute name="fromAuthorizedRoleIDRef"
4248 type="xsd:IDREF" />
4249         <xsd:attribute name="toAuthorizedRole"
4250 type="xsd:string" use="required" />
4251         <xsd:attribute name="toAuthorizedRoleIDRef"
4252 type="xsd:IDREF" />
4253         <xsd:attribute name="isConcurrent" type="xsd:boolean"
4254 value="true" />
4255         <xsd:attribute name="isLegallyBinding"
4256 type="xsd:boolean" value="true" />
4257         <xsd:attribute name="timeToPerform"
4258 type="xsd:timeDuration" />
4259     </xsd:complexType>
4260 </xsd:element>
4261 <xsd:element name="CollaborationActivity">
4262     <xsd:complexType>
4263         <xsd:sequence>
4264             <xsd:element ref="Documentation" minOccurs="0"
4265 maxOccurs="unbounded" />
4266         </xsd:sequence>
4267         <xsd:attribute name="nameID" type="xsd:ID" />
4268         <xsd:attribute name="name" type="xsd:string"
4269 use="required" />
4270         <xsd:attribute name="fromAuthorizedRole"
4271 type="xsd:string" use="required" />
4272         <xsd:attribute name="fromAuthorizedRoleIDRef"
4273 type="xsd:IDREF" />
4274         <xsd:attribute name="toAuthorizedRole"
4275 type="xsd:string" use="required" />
4276         <xsd:attribute name="toAuthorizedRoleIDRef"
4277 type="xsd:IDREF" />
4278         <xsd:attribute name="binaryCollaboration"
4279 type="xsd:string" use="required" />
4280         <xsd:attribute name="binaryCollaborationIDRef"
4281 type="xsd:IDREF" />
4282     </xsd:complexType>
4283 </xsd:element>
4284 <xsd:element name="DocumentEnvelope">
4285     <xsd:complexType>
4286         <xsd:sequence>
4287             <xsd:element ref="Documentation" minOccurs="0"
4288 maxOccurs="unbounded" />
4289             <xsd:element ref="Attachment" minOccurs="0"
4290 maxOccurs="unbounded" />
4291         </xsd:sequence>

```

```

4292         <xsd:attribute name="businessDocument"
4293 type="xsd:string" use="required"/>
4294         <xsd:attribute name="businessDocumentIDRef"
4295 type="xsd:IDREF"/>
4296         <xsd:attribute name="isPositiveResponse"
4297 type="xsd:string"/>
4298         <xsd:attribute name="isAuthenticated"
4299 type="xsd:boolean" value="false"/>
4300         <xsd:attribute name="isConfidential"
4301 type="xsd:boolean" value="false"/>
4302         <xsd:attribute name="isTamperProof"
4303 type="xsd:boolean" value="false"/>
4304     </xsd:complexType>
4305 </xsd:element>
4306
4307     <xsd:element name="Documentation">
4308         <xsd:complexType>
4309             <xsd:simpleContent>
4310                 <xsd:restriction base="xsd:string">
4311                     <xsd:attribute name="uri"
4312 type="xsd:uriReference"/>
4313                 </xsd:restriction>
4314             </xsd:simpleContent>
4315         </xsd:complexType>
4316     </xsd:element>
4317     <xsd:element name="Failure">
4318         <xsd:complexType>
4319             <xsd:sequence >
4320                 <xsd:element ref="Documentation" minOccurs="0"
4321 maxOccurs="unbounded"/>
4322                 <xsd:element ref="ConditionExpression"
4323 minOccurs="0" maxOccurs="1"/>
4324             </xsd:sequence>
4325             <xsd:attribute name="fromBusinessState"
4326 type="xsd:string" use="required"/>
4327             <xsd:attribute name="fromBusinessStateIDRef"
4328 type="xsd:IDREF"/>
4329             <xsd:attribute name="conditionGuard">
4330                 <xsd:simpleType>
4331                     <xsd:restriction base="xsd:NMTOKEN">
4332                         <xsd:enumeration value="Success"/>
4333                         <xsd:enumeration
4334 value="BusinessFailure"/>
4335                         <xsd:enumeration
4336 value="TechnicalFailure"/>
4337                         <xsd:enumeration
4338 value="AnyFailure"/>
4339                     </xsd:restriction>
4340                 </xsd:simpleType>
4341             </xsd:attribute>
4342         </xsd:complexType>
4343     </xsd:element>
4344     <xsd:element name="Fork">
4345         <xsd:complexType>
4346             <xsd:sequence>

```



```
4347             <xsd:element ref="Documentation" minOccurs="0"
4348 maxOccurs="unbounded" />
4349         </xsd:sequence>
4350         <xsd:attribute name="name" type="xsd:string"
4351 use="required" />
4352         <xsd:attribute name="nameID" type="xsd:ID" />
4353     </xsd:complexType>
4354 </xsd:element>
4355 <xsd:element name="Include">
4356     <xsd:complexType>
4357         <xsd:sequence>
4358             <xsd:element ref="Documentation" minOccurs="0"
4359 maxOccurs="unbounded" />
4360         </xsd:sequence>
4361         <xsd:attribute name="name" type="xsd:string"
4362 use="required" />
4363         <xsd:attribute name="uuid" type="xsd:string"
4364 use="required" />
4365         <xsd:attribute name="uri" type="xsd:uriReference"
4366 use="required" />
4367         <xsd:attribute name="version" type="xsd:string"
4368 use="required" />
4369     </xsd:complexType>
4370 </xsd:element>
4371 <xsd:element name="Join">
4372     <xsd:complexType>
4373         <xsd:sequence>
4374             <xsd:element ref="Documentation" minOccurs="0"
4375 maxOccurs="unbounded" />
4376         </xsd:sequence>
4377         <xsd:attribute name="name" type="xsd:string"
4378 use="required" />
4379         <xsd:attribute name="nameID" type="xsd:ID" />
4380         <xsd:attribute name="waitForAll" type="xsd:boolean"
4381 value="true" />
4382     </xsd:complexType>
4383 </xsd:element>
4384 <xsd:element name="MultiPartyCollaboration">
4385     <xsd:complexType>
4386         <xsd:sequence>
4387             <xsd:element ref="Documentation" minOccurs="0"
4388 maxOccurs="unbounded" />
4389             <xsd:element ref="BusinessPartnerRole"
4390 minOccurs="0" maxOccurs="unbounded" />
4391         </xsd:sequence>
4392         <xsd:attribute name="name" type="xsd:string"
4393 use="required" />
4394         <xsd:attribute name="nameID" type="xsd:ID" />
4395     </xsd:complexType>
4396 </xsd:element>
4397 <xsd:element name="Package">
4398     <xsd:complexType>
4399         <xsd:sequence>
4400             <xsd:element ref="Documentation" minOccurs="0"
4401 maxOccurs="unbounded" />
```

```
4402             <xsd:choice minOccurs="0"
4403 maxOccurs="unbounded">
4404                 <xsd:element ref="Package"/>
4405                 <xsd:element ref="BinaryCollaboration"/>
4406                 <xsd:element ref="BusinessTransaction"/>
4407                 <xsd:element
4408 ref="MultiPartyCollaboration"/>
4409             </xsd:choice>
4410         </xsd:sequence>
4411         <xsd:attribute name="name" type="xsd:string"
4412 use="required"/>
4413         <xsd:attribute name="nameID" type="xsd:ID"/>
4414     </xsd:complexType>
4415 </xsd:element>
4416 <xsd:element name="Performs">
4417     <xsd:complexType>
4418         <xsd:sequence>
4419             <xsd:element ref="Documentation" minOccurs="0"
4420 maxOccurs="unbounded"/>
4421         </xsd:sequence>
4422         <xsd:attribute name="initiatingRole"
4423 type="xsd:string" use="required"/>
4424         <xsd:attribute name="initiatingRoleIDRef"
4425 type="xsd:IDREF"/>
4426         <xsd:attribute name="respondingRole"
4427 type="xsd:string" use="required"/>
4428         <xsd:attribute name="respondingRoleIDRef"
4429 type="xsd:IDREF"/>
4430     </xsd:complexType>
4431 </xsd:element>
4432 <xsd:element name="ProcessSpecification">
4433     <xsd:complexType>
4434         <xsd:sequence>
4435             <xsd:element ref="Documentation" minOccurs="0"
4436 maxOccurs="unbounded"/>
4437             <xsd:element ref="SubstitutionSet"
4438 minOccurs="0" maxOccurs="unbounded"/>
4439             <xsd:choice minOccurs="0"
4440 maxOccurs="unbounded">
4441                 <xsd:element ref="Include" minOccurs="0"
4442 maxOccurs="unbounded"/>
4443                 <xsd:element ref="BusinessDocument"
4444 minOccurs="0" maxOccurs="unbounded"/>
4445                 <xsd:element ref="ProcessSpecification"
4446 minOccurs="0" maxOccurs="unbounded"/>
4447                 <xsd:element ref="Package"/>
4448                 <xsd:element ref="BinaryCollaboration"/>
4449                 <xsd:element ref="BusinessTransaction"/>
4450                 <xsd:element
4451 ref="MultiPartyCollaboration"/>
4452             </xsd:choice>
4453         </xsd:sequence>
4454         <xsd:attribute name="name" type="xsd:ID"
4455 use="required"/>
```

```
4456         <xsd:attribute name="uuid" type="xsd:string"
4457 use="required" />
4458         <xsd:attribute name="version" type="xsd:string"
4459 use="required" />
4460     </xsd:complexType>
4461 </xsd:element>
4462 <xsd:element name="RequestingBusinessActivity">
4463     <xsd:complexType>
4464         <xsd:sequence>
4465             <xsd:element ref="Documentation" minOccurs="0"
4466 maxOccurs="unbounded" />
4467             <xsd:element ref="DocumentEnvelope" />
4468         </xsd:sequence>
4469         <xsd:attribute name="name" type="xsd:string"
4470 use="required" />
4471         <xsd:attribute name="nameID" type="xsd:ID" />
4472         <xsd:attribute name="isAuthorizationRequired"
4473 type="xsd:boolean" value="false" />
4474         <xsd:attribute name="isIntelligibleCheckRequired"
4475 type="xsd:boolean" value="false" />
4476         <xsd:attribute name="isNonRepudiationReceiptRequired"
4477 type="xsd:boolean" value="false" />
4478         <xsd:attribute name="isNonRepudiationRequired"
4479 type="xsd:boolean" value="false" />
4480         <xsd:attribute name="timeToAcknowledgeAcceptance"
4481 type="xsd:timeDuration" />
4482         <xsd:attribute name="timeToAcknowledgeReceipt"
4483 type="xsd:timeDuration" />
4484     </xsd:complexType>
4485 </xsd:element>
4486 <xsd:element name="RespondingBusinessActivity">
4487     <xsd:complexType>
4488         <xsd:sequence>
4489             <xsd:element ref="Documentation" minOccurs="0"
4490 maxOccurs="unbounded" />
4491             <xsd:element ref="DocumentEnvelope"
4492 minOccurs="0" maxOccurs="unbounded" />
4493         </xsd:sequence>
4494         <xsd:attribute name="name" type="xsd:string"
4495 use="required" />
4496         <xsd:attribute name="nameID" type="xsd:ID" />
4497         <xsd:attribute name="isAuthorizationRequired"
4498 type="xsd:boolean" value="false" />
4499         <xsd:attribute name="isIntelligibleCheckRequired"
4500 type="xsd:boolean" value="false" />
4501         <xsd:attribute name="isNonRepudiationReceiptRequired"
4502 type="xsd:boolean" value="false" />
4503         <xsd:attribute name="isNonRepudiationRequired"
4504 type="xsd:boolean" value="false" />
4505         <xsd:attribute name="timeToAcknowledgeReceipt"
4506 type="xsd:timeDuration" />
4507     </xsd:complexType>
4508 </xsd:element>
4509 <xsd:element name="Start">
4510     <xsd:complexType>
```

```
4511         <xsd:sequence>
4512             <xsd:element ref="Documentation" minOccurs="0"
4513 maxOccurs="unbounded" />
4514         </xsd:sequence>
4515         <xsd:attribute name="toBusinessState"
4516 type="xsd:string" use="required" />
4517         <xsd:attribute name="toBusinessStateIDRef"
4518 type="xsd:IDREF" />
4519     </xsd:complexType>
4520 </xsd:element>
4521 <xsd:element name="Success">
4522     <xsd:complexType>
4523         <xsd:sequence >
4524             <xsd:element ref="Documentation" minOccurs="0"
4525 maxOccurs="unbounded" />
4526             <xsd:element ref="ConditionExpression"
4527 minOccurs="0" maxOccurs="1" />
4528         </xsd:sequence>
4529         <xsd:attribute name="fromBusinessState"
4530 type="xsd:string" use="required" />
4531         <xsd:attribute name="fromBusinessStateIDRef"
4532 type="xsd:IDREF" />
4533         <xsd:attribute name="conditionGuard">
4534             <xsd:simpleType>
4535                 <xsd:restriction base="xsd:NMTOKEN">
4536                     <xsd:enumeration value="Success" />
4537                     <xsd:enumeration
4538 value="BusinessFailure" />
4539                     <xsd:enumeration
4540 value="TechnicalFailure" />
4541                     <xsd:enumeration
4542 value="AnyFailure" />
4543                 </xsd:restriction>
4544             </xsd:simpleType>
4545         </xsd:attribute>
4546     </xsd:complexType>
4547 </xsd:element>
4548 <xsd:element name="Transition">
4549     <xsd:complexType>
4550         <xsd:sequence >
4551             <xsd:element ref="Documentation" minOccurs="0"
4552 maxOccurs="unbounded" />
4553             <xsd:element ref="ConditionExpression"
4554 minOccurs="0" maxOccurs="1" />
4555         </xsd:sequence>
4556         <xsd:attribute name="onInitiation" type="xsd:boolean"
4557 value="false" />
4558         <xsd:attribute name="fromBusinessState"
4559 type="xsd:string" />
4560         <xsd:attribute name="fromBusinessStateIDRef"
4561 type="xsd:IDREF" />
4562         <xsd:attribute name="toBusinessState"
4563 type="xsd:string" />
4564         <xsd:attribute name="toBusinessStateIDRef"
4565 type="xsd:IDREF" />
```

```
4566         <xsd:attribute name="conditionGuard">
4567             <xsd:simpleType>
4568                 <xsd:restriction base="xsd:NMTOKEN">
4569                     <xsd:enumeration value="Success"/>
4570                     <xsd:enumeration
4571 value="BusinessFailure"/>
4572                     <xsd:enumeration
4573 value="TechnicalFailure"/>
4574                     <xsd:enumeration
4575 value="AnyFailure"/>
4576                 </xsd:restriction>
4577             </xsd:simpleType>
4578         </xsd:attribute>
4579     </xsd:complexType>
4580 </xsd:element>
4581 </xsd:schema>
4582
4583
```

4583 **11 References**

4584

4585

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

4586

2. RosettaNet Implementation Framework: Core Specification, Version:

4587

Release 2.00.00, 3 January 2001

4588

4589 **12 Disclaimer**

4590

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

4591

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

4592

employers specifically disclaim responsibility for any problems arising from

4593

correct or incorrect implementation or use of this design.

4594

4594 **13 Contact Information**

4595

4596

Team Leader (Of the BP team):

4597

Paul Levine

4598

Telcordia Technologies, Inc.

4599

45 Knightsbridge Road

4600

Piscataway, N.J. 08854

4601

US

4602

4603

Phone: 732-699-3042

4604

EMail: plevine@telcordia.com

4605

4606

Sub Team Lead (Of the context/MetamodelGroup) :

4607

4608

Karsten Riemer

4609

Sun Microsystems

4610

1 Network Drive

4611

Burlington, MA 01803

4612

USA

4613

4614

Phone: 781-442-2679

4615

EMail: karsten.riemer@sun.com

4616

4617

Editor (of this document):

4618

4619

Karsten Riemer

4620

Sun Microsystems

4621

1 Network Drive

4622

Burlington, MA 01803

4623

USA

4624

4625

Phone: 781-442-2679

4626

EMail: karsten.riemer@sun.com

4627

4628 Copyright Statement

4629 Copyright © UN/CEFACT and OASIS, 2001. All Rights Reserved.

4630

4631 This document and translations of it may be copied and furnished to others, and
4632 derivative works that comment on or otherwise explain it or assist in its implementation
4633 may be prepared, copied, published and distributed, in whole or in part, without
4634 restriction of any kind, provided that the above copyright notice and this paragraph are
4635 included on all such copies and derivative works. However, this document itself may not
4636 be modified in any way, such as by removing the copyright notice or references to
4637 ebXML, UN/CEFACT, or OASIS, except as required to translate it into languages other
4638 than English.

4639

4640 The limited permissions granted above are perpetual and will not be revoked by ebXML
4641 or its successors or assigns. This document and the information contained herein is
4642 provided on an "AS IS" basis and ebXML DISCLAIMS ALL WARRANTIES,
4643 EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY
4644 THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY
4645 RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR
4646 FITNESS FOR A PARTICULAR PURPOSE

4647