Universal Business Language Version 2.1

Committee Specification Draft 03 / Public Review Draft 03 *** sanity check 1 -- edit in progress -- absent latest SGTG deliverables ***

03 November 2012 02:10z

Specification URIs

This version:

http://docs.oasis-open.org/ubl/prd3-UBL-2.1/UBL-2.1.html
http://docs.oasis-open.org/ubl/prd3-UBL-2.1/UBL-2.1.pdf
http://docs.oasis-open.org/ubl/prd3-UBL-2.1/UBL-2.1.xml (Authoritative)

Previous version:

http://docs.oasis-open.org/ubl/prd2-UBL-2.1/UBL-2.1.html
http://docs.oasis-open.org/ubl/prd2-UBL-2.1/UBL-2.1.pdf
http://docs.oasis-open.org/ubl/prd2-UBL-2.1/UBL-2.1.xml

Latest version:

http://docs.oasis-open.org/ubl/UBL-2.1.html
http://docs.oasis-open.org/ubl/UBL-2.1.pdf
http://docs.oasis-open.org/ubl/UBL-2.1.xml (Authoritative)

Technical Committee:

OASIS Universal Business Language TC

Chairs:

Jon Bosak (bosak@pinax.com), Pinax
Tim McGrath (tim.mcgrath@documentengineeringservices.com), Document Engineering Services

Editors:

Jon Bosak (bosak@pinax.com), Pinax
Tim McGrath (tim.mcgrath@documentengineeringservices.com), Document Engineering Services
G. Ken Holman (gkholman@CraneSoftwrights.com), Crane Softwrights Ltd.

Related Work:

This specification supersedes UBL 2.0.

Declared XML Namespaces:

urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2
urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2
urn:oasis:names:specification:ubl:schema:xsd:CommonExtensionComponents-2
urn:oasis:names:specification:ubl:schema:xsd:CommonSignatureComponents-2
urn:oasis:names:specification:ubl:schema:xsd:QualifiedDataTypes-2

urn:oasis:names:specification:ubl:schema:xsd:SignatureAggregateComponents-2
urn:oasis:names:specification:ubl:schema:xsd:SignatureBasicComponents-2
urn:oasis:names:specification:ubl:schema:xsd:UnqualifiedDataTypes-2

urn:oasis:names:specification:ubl:schema:xsd:ApplicationResponse-2
urn:oasis:names:specification:ubl:schema:xsd:AttachedDocument-2
urn:oasis:names:specification:ubl:schema:xsd:AwardedNotification-2
urn:oasis:names:specification:ubl:schema:xsd:BillOfLading-2
urn:oasis:names:specification:ubl:schema:xsd:CallForTenders-2
urn:oasis:names:specification:ubl:schema:xsd:Catalogue-2
urn:oasis:names:specification:ubl:schema:xsd:CatalogueDeletion-2
urn:oasis:names:specification:ubl:schema:xsd:CatalogueItemSpecificationUpdate-2
urn:oasis:names:specification:ubl:schema:xsd:CataloguePricingUpdate-2
urn:oasis:names:specification:ubl:schema:xsd:CatalogueRequest-2
urn:oasis:names:specification:ubl:schema:xsd:CertificateOfOrigin-2
urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2
urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2
urn:oasis:names:specification:ubl:schema:xsd:CommonExtensionComponents-2
urn:oasis:names:specification:ubl:schema:xsd:ContractAwardNotice-2
urn:oasis:names:specification:ubl:schema:xsd:ContractNotice-2
urn:oasis:names:specification:ubl:schema:xsd:CreditNote-2
urn:oasis:names:specification:ubl:schema:xsd:DebitNote-2
urn:oasis:names:specification:ubl:schema:xsd:DespatchAdvice-2
urn:oasis:names:specification:ubl:schema:xsd:DocumentStatus-2
urn:oasis:names:specification:ubl:schema:xsd:DocumentStatusRequest-2
urn:oasis:names:specification:ubl:schema:xsd:ExceptionCriteria-2
urn:oasis:names:specification:ubl:schema:xsd:ExceptionNotification-2
urn:oasis:names:specification:ubl:schema:xsd:Forecast-2
urn:oasis:names:specification:ubl:schema:xsd:ForecastRevision-2
urn:oasis:names:specification:ubl:schema:xsd:ForwardingInstructions-2
urn:oasis:names:specification:ubl:schema:xsd:FreightInvoice-2
urn:oasis:names:specification:ubl:schema:xsd:GuaranteeCertificate-2
urn:oasis:names:specification:ubl:schema:xsd:InstructionForReturns-2
urn:oasis:names:specification:ubl:schema:xsd:InventoryReport-2
urn:oasis:names:specification:ubl:schema:xsd:Invoice-2
urn:oasis:names:specification:ubl:schema:xsd:ItemInformationRequest-2
urn:oasis:names:specification:ubl:schema:xsd:Order-2
urn:oasis:names:specification:ubl:schema:xsd:OrderCancellation-2
urn:oasis:names:specification:ubl:schema:xsd:OrderChange-2
urn:oasis:names:specification:ubl:schema:xsd:OrderResponse-2
urn:oasis:names:specification:ubl:schema:xsd:OrderResponseSimple-2
urn:oasis:names:specification:ubl:schema:xsd:PackingList-2
urn:oasis:names:specification:ubl:schema:xsd:PerformanceHistory-2
urn:oasis:names:specification:ubl:schema:xsd:PriorInformationNotice-2
urn:oasis:names:specification:ubl:schema:xsd:ProductActivity-2
urn:oasis:names:specification:ubl:schema:xsd:Quotation-2
urn:oasis:names:specification:ubl:schema:xsd:ReceiptAdvice-2
urn:oasis:names:specification:ubl:schema:xsd:Reminder-2
urn:oasis:names:specification:ubl:schema:xsd:RemittanceAdvice-2
urn:oasis:names:specification:ubl:schema:xsd:RequestForQuotation-2
urn:oasis:names:specification:ubl:schema:xsd:RetailEvent-2
urn:oasis:names:specification:ubl:schema:xsd:SelfBilledCreditNote-2
urn:oasis:names:specification:ubl:schema:xsd:SelfBilledInvoice-2
urn:oasis:names:specification:ubl:schema:xsd:Statement-2
urn:oasis:names:specification:ubl:schema:xsd:StockAvailabilityReport-2
urn:oasis:names:specification:ubl:schema:xsd:Tender-2
urn:oasis:names:specification:ubl:schema:xsd:TendererQualification-2
urn:oasis:names:specification:ubl:schema:xsd:TendererQualificationResponse-2

urn:oasis:names:specification:ubl:schema:xsd:TenderReceipt-2
urn:oasis:names:specification:ubl:schema:xsd:TradeItemLocationProfile-2
urn:oasis:names:specification:ubl:schema:xsd:TransportationStatus-2
urn:oasis:names:specification:ubl:schema:xsd:TransportationStatusRequest-2
urn:oasis:names:specification:ubl:schema:xsd:TransportExecutionStatus-2
urn:oasis:names:specification:ubl:schema:xsd:TransportProgressStatus-2
urn:oasis:names:specification:ubl:schema:xsd:UnawardedNotification-2
urn:oasis:names:specification:ubl:schema:xsd:UtilityStatement-2
urn:oasis:names:specification:ubl:schema:xsd:Waybill-2

Abstract:

This specification defines the Universal Business Language, version 2.1.

Status:

This document was last revised or approved by the UBL TC on the above date. The level of approval is also listed above. Check the current location noted above for possible later revisions of this document. This document is updated periodically on no particular schedule.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee's web page at http://www.oasis-open.org/committees/ubl/.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page at http://www.oasis-open.org/committees/ubl/ipr.php.

See Appendix A, *Release Notes (Non-Normative)* for more information regarding this release package.

Citation format:

When referencing this specification the following citation format should be used:

[**UBL-2.1**] *Universal Business Language Version 2.1 03 November 2012 02:10z. Committee Specification Draft 03 / Public Review Draft 03. http://docs.oasis-open.org/ubl/prd3-UBL-2.1/UBL-2.1.html*

Notices

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the “OASIS IPR Policy”). The full Policy may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an “AS IS” basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS’ procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name “OASIS” is a trademark of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see http://www.oasis-open.org/who/trademark.php for above guidance.

CPFR is a registered trademark of the Voluntary Interindustry Commerce Solutions (VICS) association. The term “CPFR” is not be used in any format without written permission from VICS. For more information on VICS and CPFR, visit www.vics.org.

Table of Contents

Appendixes

1 Introduction

Since its approval as a W3C recommendation in 1998, XML has been adopted in a number of industries as a framework for the definition of the messages exchanged in electronic commerce. The widespread use of XML has led to the development of multiple industry-specific XML versions of such basic documents as purchase orders, shipping notices, and invoices.

While industry-specific data formats have the advantage of maximal optimization for their business context, the existence of different formats to accomplish the same purpose in different business domains is attended by a number of significant disadvantages as well.

- Developing and maintaining multiple versions of common business documents like purchase orders and invoices is a major duplication of effort.
- Creating and maintaining multiple adapters to enable trading relationships across domain boundaries is an even greater effort.
- The existence of multiple XML formats makes it much harder to integrate XML business messages with back-office systems.
- The need to support an arbitrary number of XML formats makes tools more expensive and trained workers harder to find.

The OASIS Universal Business Language (UBL) is intended to help solve these problems by defining a generic XML interchange format for business documents that can be extended to meet the requirements of particular industries. Specifically, UBL provides the following:

- A suite of structured business objects and their associated semantics expressed as reusable data components and common business documents.
- A library of XML schemas for reusable data components such as “Address”, “Item”, and “Payment”—the common data elements of everyday business documents.
- A set of XML schemas for common business documents such as “Order”, “Despatch Advice”, and “Invoice” that are constructed from the UBL library components and can be used in generic procurement and transportation contexts.

A standard basis for XML business schemas provides the following advantages:

- Lower cost of integration, both among and within enterprises, through the reuse of common data structures.
- Lower cost of commercial software, because software written to process a given XML tag set is much easier to develop than software that can handle an unlimited number of tag sets.
- An easier learning curve, because users need master just a single library.
- Lower cost of entry and therefore quicker adoption by small and medium-size enterprises (SMEs).
- Standardized training, resulting in many skilled workers.
- A universally available pool of system integrators.
- Standardized, inexpensive data input and output tools.
- A standard target for inexpensive off-the-shelf business software.

UBL is designed to provide a universally understood and recognized commercial syntax for legally binding business documents and to operate within a standard business framework such as ISO 15000

(ebXML) to provide a complete, standards-based infrastructure that can extend the benefits of existing EDI systems to businesses of all sizes. UBL is freely available to everyone without legal encumbrance or licensing fees.

UBL schemas are modular, reusable, and extensible in XML-aware ways. As the first standard implementation of ebXML Core Components Technical Specification 2.01, the UBL Library is based on a conceptual model of information components known as Business Information Entities (BIEs). These components are assembled into specific document models such as Order and Invoice. These document assembly models are then transformed in accordance with UBL Naming and Design Rules into W3C XSD schema syntax. This approach facilitates the creation of UBL-based document types beyond those specified in this release.

UBL can be regarded as a generic Open-edi Configuration in the perspective of the Open-edi reference model (ISO/IEC 14662:2010). This is described in more detail in Appendix I, *The Open-edi reference model perspective of UBL (Non-Normative)*.

1.1 Terminology

1.1.1 Terms and Definitions

ASiC-S
Associated Signature Container (simple form). A standard container that associates a single data object with one or more detached signature(s) that apply to it. See [**ASiC**].

Assembly model
A tree-structured model of ABIEs that can be implemented as a document schema. In this release, assembly models are provided in tabular form as spreadsheets. Each spreadsheet is included in the package in two formats, Open Document format and Microsoft Excel format, with identical cell content. See Appendix C, *The UBL 2.1 Data Model (Non-Normative)* for more about UBL document assembly.

Digital Signature
A value generated from the application of a private key to a message via a cryptographic algorithm such that it has the properties of integrity and message authentication and/or signer authentication. A signature may be (non-exclusively) described as detached, enveloping, or enveloped ([**xmldsig**], with modifications).

Document
A set of information components that are interchanged as part of a business transaction; for example, in placing an order.

Transform
The processing of data from its source to its derived form. Typical transforms include XML Canonicalization [**XML C14N**] and XSLT [**XSLT 2.0**].

XSD schema
An XML document definition conforming to the W3C XML Schema language [**XSD1**][**XSD2**].

The terms *Core Component (CC), Basic Core Component (BCC), Aggregate Core Component (ACC), Association Core Component (ASCC), Business Information Entity (BIE), Basic Business Information Entity (BBIE),* and *Aggregate Business Information Entity (ABIE)* are used in this specification with the meanings given in [**CCTS**].

The terms *Object Class, Property Term, Representation Term,* and *Qualifier* are used in this specification with the meanings given in [**ISO11179**].

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY and OPTIONAL, when they appear in this document, are to be interpreted as described in [**RFC2119**].

1.1.2 Symbols and Abbreviations

ABIE
Aggregate Business Information Entity

AdES
Advanced Electronic Signature

ASBIE
Association Business Information Entity

BBIE
Basic Business Information Entity

BIE
Business Information Entity

C14N
Canonicalization

CC
Core Component

CPFR
Collaborative Planning, Forecasting, and Replenishment [**CPFR**]

CV2
Credit Card Verification Numbering System

DSig
Digital Signature

EDI
Electronic Data Interchange

ISO
International Organization for Standardization

NDR
UBL Naming and Design Rules

QC
Qualified Certificate

QS
Qualified Signature

UML
Unified Modeling Language [**UML**]

UN/CEFACT
United Nations Centre for Trade Facilitation and Electronic Business

UNDG
United Nations Dangerous Goods

URI
Uniform Resource Identifier

UUID
Universally Unique Identifier

XAdES
XML Advanced Electronic Signatures [**XAdES**]

XML
Extensible Markup Language [**XML**]

XMLDSig
XML Digital Signature [**xmldsig**]

XPath
The XML Path Language

XSD
W3C XML Schema Language [**XSD1**][**XSD2**]

XSLT
Extensible Stylesheet Language Transformations (a transformation language) [**XSLT 2.0**]

1.2 Normative References

[**ASN.1**] *ITU-T X.680-X.683: Abstract Syntax Notation One (ASN.1) [http://www.itu.int/ITU-T/studygroups/com17/languages/X.680-X.693-0207w.zip]*, *ITU-T X.690-X.693: ASN.1 encoding rules [http://www.oasis-open.org/committees/download.php/6320/X.680-X.693-0207w.zip]*

[**CCTS**] *ISO/TS 15000-5:2005 Electronic Business Extensible Markup Language (ebXML)— Part 5: ebXML Core Components Technical Specification, Version 2.01 [http://www.oasis-open.org/committees/download.php/6232/CEFACT-CCTS-Version-2pt01.zip] (identical to Part 8 of the ebXML Framework)*

[**CPFR**] *Voluntary Interindustry Commerce Standards, Collaborative Planning, Forecasting, and Replenishment Version 2.0, Global Commerce Initiative Recommended Guidelines, June 2002 [http://www.vics.org/docs/committees/cpfr/CPFR_Tabs_061802.pdf]*

[**Customization**] *OASIS Committee Specification 01, UBL 2 Guidelines for Customization, First Edition, 25 December 2009 [http://docs.oasis-open.org/ubl/guidelines/UBL2-Customization1.0cs01.pdf]*

[**CVA**] *OASIS Committee Specification 01, Context/value association using Genericode 1.0, 15 April 2010 [http://docs.oasis-open.org/codelist/cs01-ContextValueAssociation-1.0/doc/context-value-association.html]*

[**genericode**] *OASIS Committee Specification 01, Code List Representation (Genericode) Version 1.0, 28 December 2007 [http://docs.oasis-open.org/codelist/cs-genericode-1.0/doc/oasis-code-list-representation-genericode.html]*

[**ISO11179**] *ISO/IEC 11179-1:1999 Information technology — Specification and standardization of data elements — Part 1: Framework for the specification and standardization of data elements [http://www.oasis-open.org/committees/download.php/6233/c002349_ISO_IEC_11179-1_1999%28E%29.pdf]*

[**RELAX NG**] *ISO/IEC 19757-2, Information technology — Document Schema Definition Language (DSDL) — Part 2: Regular-grammar-based validation — RELAX NG [http://standards.iso.org/ittf/PubliclyAvailableStandards/c037605_ISO_IEC_19757-2_2003(E).zip]* , *Information technology — Document Schema Definition Language (DSDL) — Part 2: Regular-grammar-based validation — RELAX NG AMENDMENT 1: Compact Syntax [http://standards.iso.org/ittf/PubliclyAvailableStandards/c040774_ISO_IEC_19757-2_2003_Amd_1_2006(E).zip]*

[**RFC2119**] *Key words for use in RFCs to Indicate Requirement Levels [http://www.faqs.org/rfcs/rfc2119.html]*

[**SCH**] *Document Schema Definition Languages (DSDL) — Part 3: Rule-based validation (Schematron) [http://standards.iso.org/ittf/PubliclyAvailableStandards/c040833_ISO_IEC_19757-3_2006(E).zip]*

[**UML**] *Unified Modeling Language Version 1.5 (formal/03-03-01) [http://www.oasis-open.org/committees/download.php/6240/03-03-01.zip]*

[**XAdES**] *XML Advanced Electronic Signatures. ETSI TS 101 903 V1.4.1, June 2009 [http://uri.etsi.org/01903/v1.4.1/ts_101903v010401p.pdf].*

[**XML**] *Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation 6 October 2000 [http://www.w3.org/TR/2000/REC-xml-20001006]*

[**xmldsig**] *XML-Signature Syntax and Processing. W3C Recommendation 12 February 2002 [http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/]*

[**XSD1**] *XML Schema Part 1: Structures. Second Edition. W3C Recommendation 28 October 2004 [http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/]*

[**XSD2**] *XML Schema Part 2: Datatypes. Second Edition. W3C Recommendation 28 October 2004 [http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/]*

[**XSLT**] *XSL Transformations (XSLT) Version 1.0, W3C Recommendation 16 November 1999 [http://www.w3.org/TR/1999/REC-xslt-19991116]*

1.3 Non-Normative References

[**1999/93/EC**] *Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures [http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31999L0093:EN:NOT].*

[**2011/130/UE**] *Commission Decision 2011/130/UE of the European Commission of 25 February 2011 on establishing minimum requirements for the cross-border processing of documents signed electronically by competent authorities under Directive 2006/123/EC of the European Parliament and of the Council on services in the internal market [http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2011:053:0066:01:EN:HTML].*

[**ASiC**] *Electronic Signatures and Infrastructures (ESI); Associated Signature Containers (ASiC). ETSI TS 102 918 V1.1.1, April 2011 [http://pda.etsi.org/exchangefolder/ts_102918v010101p.pdf].*

[**BOV-FSV**] *ISO/IEC 15944-20 Information technology - Business operational view - Linking business operational view to functional service view*

[**CPFRoverview**] *CPFR: An Overview, 18 May 2004 [http://www.vics.org/docs/standards/CPFR_Overview_US-A4.pdf]*

[**CWA15579**] *CEN Workshop Agreement: E-invoices and digital signatures (CWA 15579), July 2006 [ftp://ftp.cenorm.be/PUBLIC/CWAs/e-Europe/eInvoicing/CWA15579-00-2006-Jul.pdf].*

[**CWA15580**] *CEN Workshop Agreement: Storage of Electronic Documents (CWA 15580), July 2006 [ftp://ftp.cenorm.be/PUBLIC/CWAs/e-Europe/eInvoicing/CWA15580-00-2006-jul.pdf].*

[**eBiz-TCF**] *Reference Architecture of eBusiness in Textile Clothing and Footwear Sector [http://spring.bologna.enea.it/ebiz/defaultebiz.asp?versione=DOWNSTREAM]*

[**Open-edi**] *ISO/IEC 14662:2010 Information technology - Open-edi reference model [http://standards.iso.org/ittf/PubliclyAvailableStandards/c055290_ISO_IEC_14662_2010%28E%29.zip]*

[**ODFP**] *OASIS Standard, Open Document Format for Office Applications (OpenDocument) Version 1.2 - Part 3 Packages, December 2006 [http://docs.oasis-open.org/office/v1.2/csprd03/OpenDocument-v1.2-csprd03-part3.pdf].*

[**RFC3161**] *Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP), August 2001 [http://www.faqs.org/rfcs/rfc3161.html].*

[**XML C14N**] John Boyer, *Canonical XML Version 1.0, 15 March 2001 [http://www.w3.org/TR/2001/REC-xml-c14n-20010315].*

[**UN/CEFACT Rec. 37**] *Signed Digital Evidence Interoperability Recommendation, 27 September 2010 [http://www.unece.org/cefact/cf_plenary/plenary10/ECE_TRADE_C_CEFACT_2010_14E.pdf].*

[**XPath 2.0**] Anders Berglund, et al., *XML Path Language (XPath) Version 2.0, 23 January 2007 [http://www.w3.org/TR/2007/REC-xpath20-20070123/].*

[**XPointer**] Steven DeRose, et al., *XML Pointer Language (XPointer) Version 1.0 Working Draft, 16 August 2002 [http://www.w3.org/TR/xptr/].*

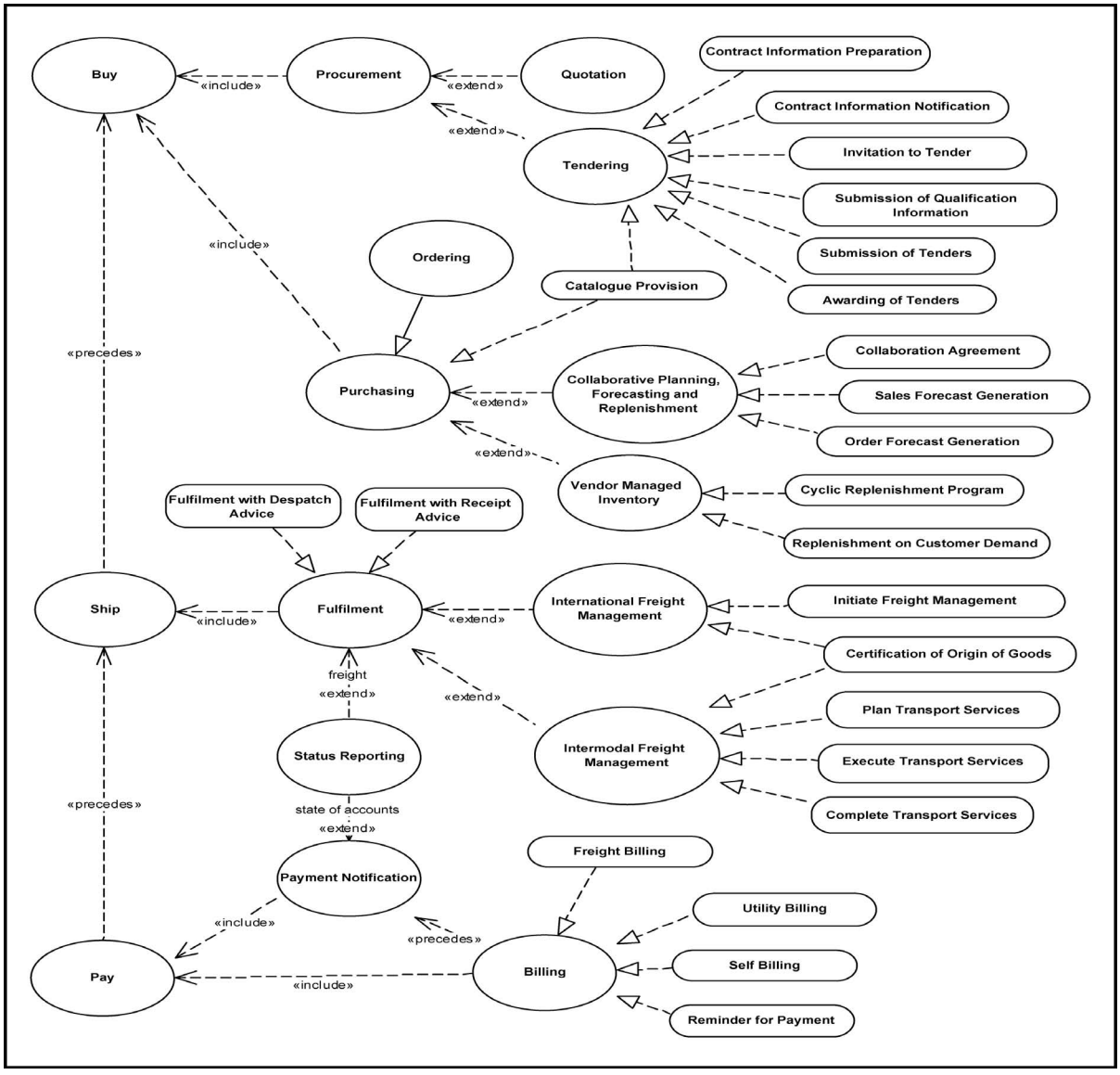
[**XSLT 2.0**] Michael Kay, *XSL Transformations (XSLT) Version 2.0, 2007-01-23 [http://www.w3.org/TR/xslt20/].*

2 UBL 2.1 Business objects

The processes described in this section, and the business rules associated with them, define a context for the use of UBL 2.1 business documents. They are normative insofar as they provide semantics for the UBL document schemas, but they should not be construed as limiting the application of those schemas.

UBL 2.1 extends the supply chain (including the commercial collaborations of international trade) to include documents for collaborative forecasting, planning and scheduling; vendor managed inventory; utility billing; tendering; and intermodal freight management. The following diagram illustrates the process context assumed by UBL 2.1 documents.

Figure 1. UBL 2.1 Use Case



The document types included in UBL 2.1 are listed in Section 3, “UBL 2.1 Schemas”. It is important to note that, as with previous UBL releases, the UBL 2.1 library is designed to support the construction of a wide variety of document types beyond those provided in the 2.1 package. It is expected that implementers will develop their own customized document types and components and that more UBL document types will be added as the library evolves. For guidance in customizing UBL document types, see the UBL Guidelines for Customization [**Customization**].

2.1 General Business Requirements

This section describes some of the requirements and general business rules that are assumed for collaborations and document exchanges in UBL 2.1.

2.1.1 Items

- An item may be a product or a service
- Items may have multiple classifications
- A contract may influence prices of items
- An item may be part of another item
- An item may have a price per unit and an order unit
- An item may reference pictures and documents
- An item may have a validity period
- An item may refer to other relevant or necessary items

Note

For a discussion of the difference between *item* and *line item* see Section D.1, “Item vs. Line Item”.)

2.1.2 Item Identification

One of the following identifiers may be used to identify each Item (for example, a product):

- Buyer’s Item Identification, or
- Seller’s Item Identification, or
- Manufacturer’s Item Identification, or
- Catalogue Item Identification, or
- Item Identification according to a system promulgated by a standards body.

The Item may be further distinguished by the specification of Measurement(s) or Physical Attribute(s). This enables specification of the following kinds of item:

- Item Requiring Description

  This is an item that is not identified by an unambiguous machine-processable product code and requires additional descriptive information to precisely identify it.

- Customer Defined Item

  This is an item that the customer describes according to his need, and in the specification of which the customer may make some reference to comparable “standard” items.

- Item Requiring Measurements

  This is an item for which it is necessary to specify one or more measurements as part of the descriptive specification of the item.

2.1.3 Item Instances

Certain Items may be identified and ordered as individual, unique objects—for example, a specific car rather than a make and model of a car. This form of identification may also be needed for product tracing (e.g., perishable goods) or because of the nature of the commodity (e.g., used, collectible, specialized, or rare).

In data modeling terms, an Item Instance is an extension of an Item.

2.1.4 Item Pricing

For any given Item, price ranges by amount, quantity, location, etc., are specified by the Seller during the sourcing stage. They are not repeated back to the Seller during Ordering; only the active price is specified.

In some cases, the Buyer may not know the Item Price, in which case it is not specified. This makes a detailed response from the Seller necessary; see Section 2.6.3, “Order Response”.

2.1.5 Hazardous Items

Although ordered items may include Hazardous items, it is not necessary to specify information related to Hazardous status at the order stage. The Buyer may not be aware of the nature of the Item. Indication of the Hazardous nature of the Item, and any relevant information, would be indicated in the Despatch Advice and Transportation documents.

2.1.6 Parties

In UBL, a party is defined as an individual, a group, or a body having a role in a business function. Dependent on the business process, a Party may play various roles in the document exchange. For a list of UBL parties and their roles, see Section 2.19, “Party Roles”.

2.1.7 Multilingual Text

Some textual components, such as Notes and Description, may be specified in several languages. Each should be a separate occurrence of the component, using the language attribute to define its presentation. However, multiple occurrences of the same textual components should not be in the same language.

2.1.8 Taxation Rules

UBL does not provide documents for tax reporting purposes. Instead, it provides structures to support the information on which taxes are based. These aim to be generic and not based on any specific tax regime.

2.2 Overview of Business Processes

Following from UBL 2.0, the UBL 2.1 documents and library support an increased range of different business processes. These processes (with the additions in 2.1 shown in underlined boldface) can be categorized as follows:

- Procurement
  - Sourcing
    - Pre-Award
      - **Tendering**
      - Catalogue
    - Post-Award
      - **Catalogue**
      - **Quotation**

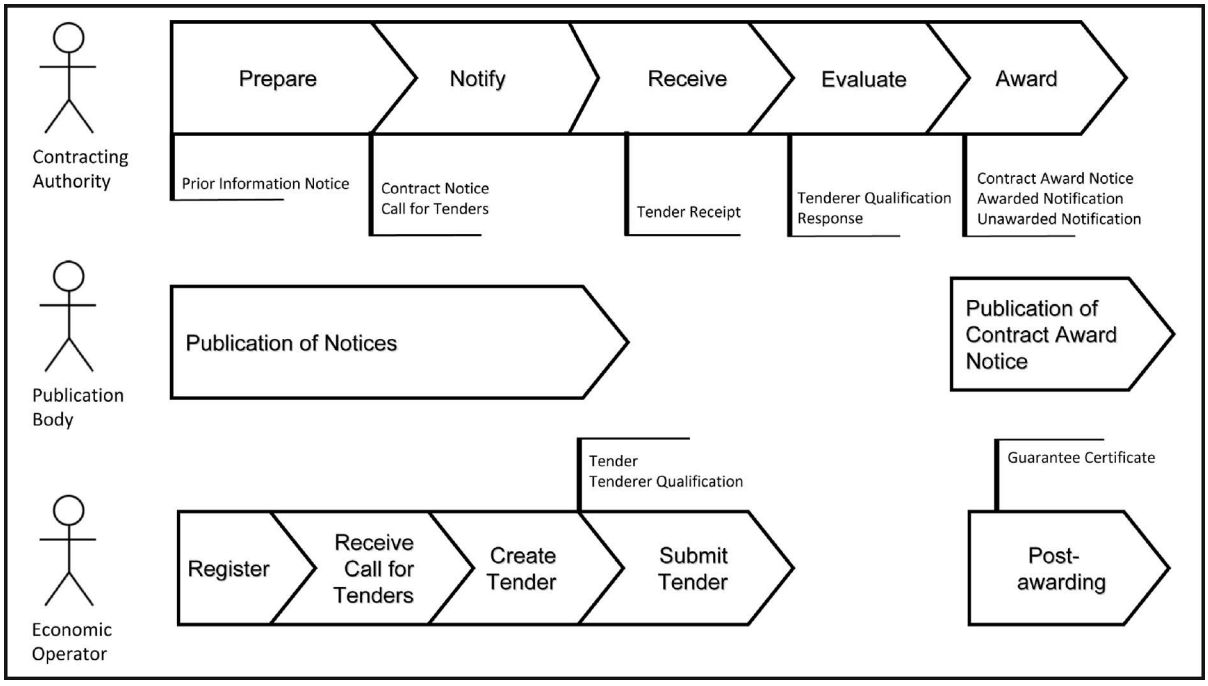
Ordering
Fulfilment
Billing
Freight Billing
Utility Billing
Payment
Replenishment
Collaborative Planning, Forecasting, and Replenishment
Vendor Managed Inventory
Cyclic Replenishment Program
Transportation
International Freight Management
Intermodal Freight Management
Freight Status Reporting
International Trade
Certification of Origin of Goods

The following sections contain the formal business process descriptions:

2.3 Tendering

Tendering is the case where a contracting authority (the Originator) initiates a procurement project to buy goods, services, or works during a specified period, as shown in the following diagram.

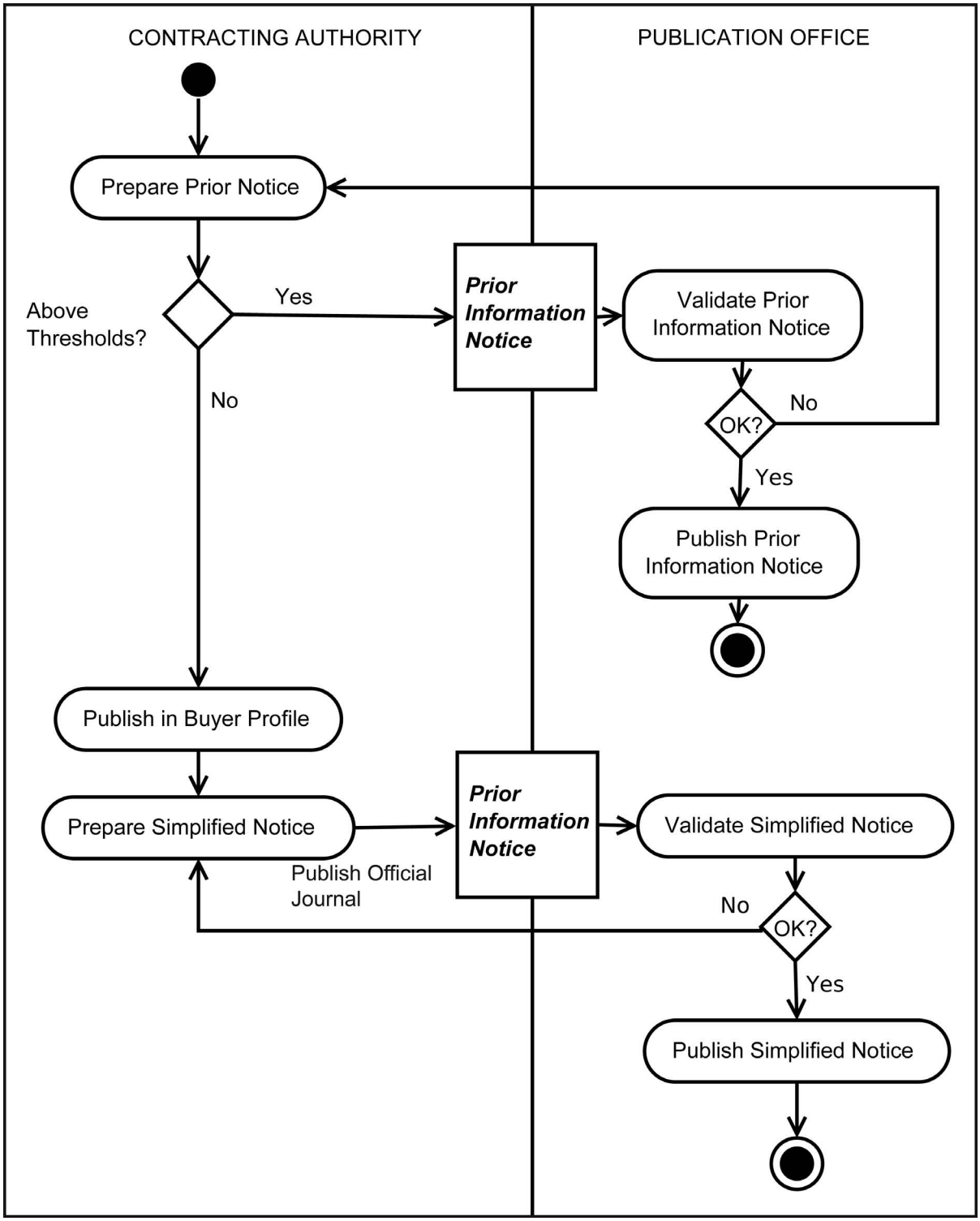
Figure 2. The Tendering Process



2.3.1 Contract Information Preparation

The Tendering process optionally begins with publication of a Prior Information Notice prepared by a Contracting Authority to *declare the intention* to buy goods, services, or works during a specified period. The purpose of this step (if implemented) is to reduce preparation time when an actual Contract Notice is published (see Section 2.3.2, “Contract Information Notification”).

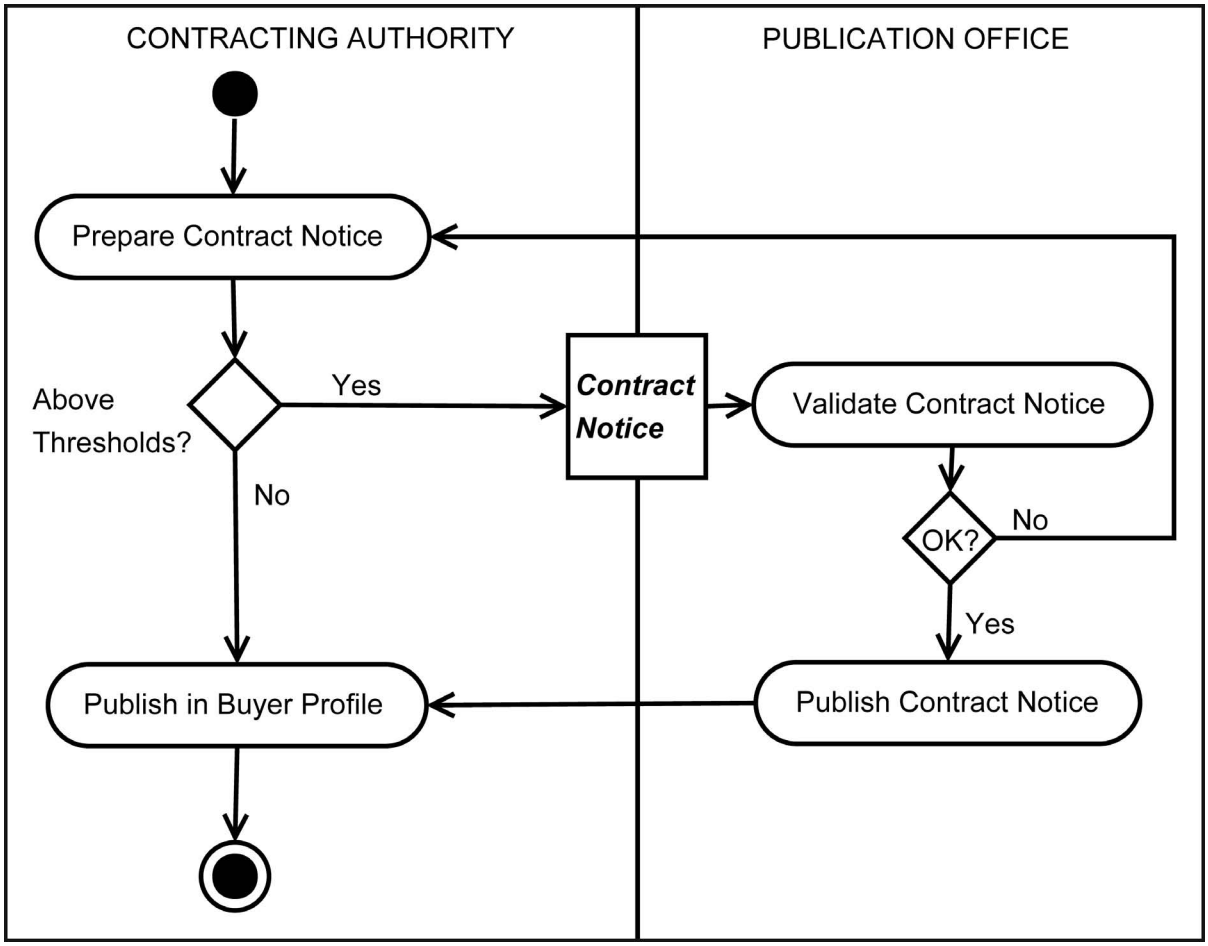
Figure 3. Contract Information Preparation



2.3.2 Contract Information Notification

The process of Notification includes the publication by the Contracting Authority of a Contract Notice to *announce* the project to buy goods, services, or works. The details shown here are specific to the EU, which requires contracts over a certain amount (Harmonized contracts) to be published in the Official Journal of the EU. Other tendering contexts will differ in their publication requirements.

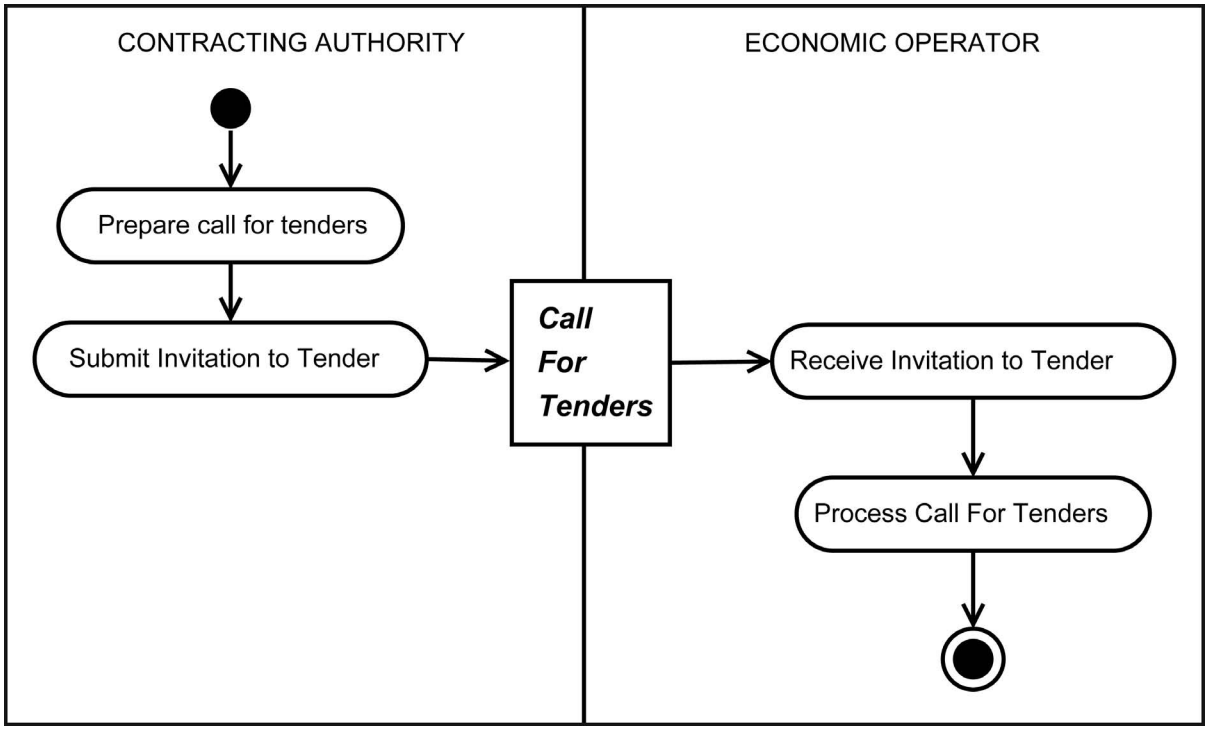
Figure 4. Contract Information Notification



2.3.3 Invitation to Tender

In some procedures, the Contracting Authority invites economic operators to participate in a contest by sending them an invitation to tender using a Call for Tenders to *define* the procurement project to buy goods, services, or works during a specified period. The Call for Tenders may be sent jointly with an unstructured letter of invitation to tender.

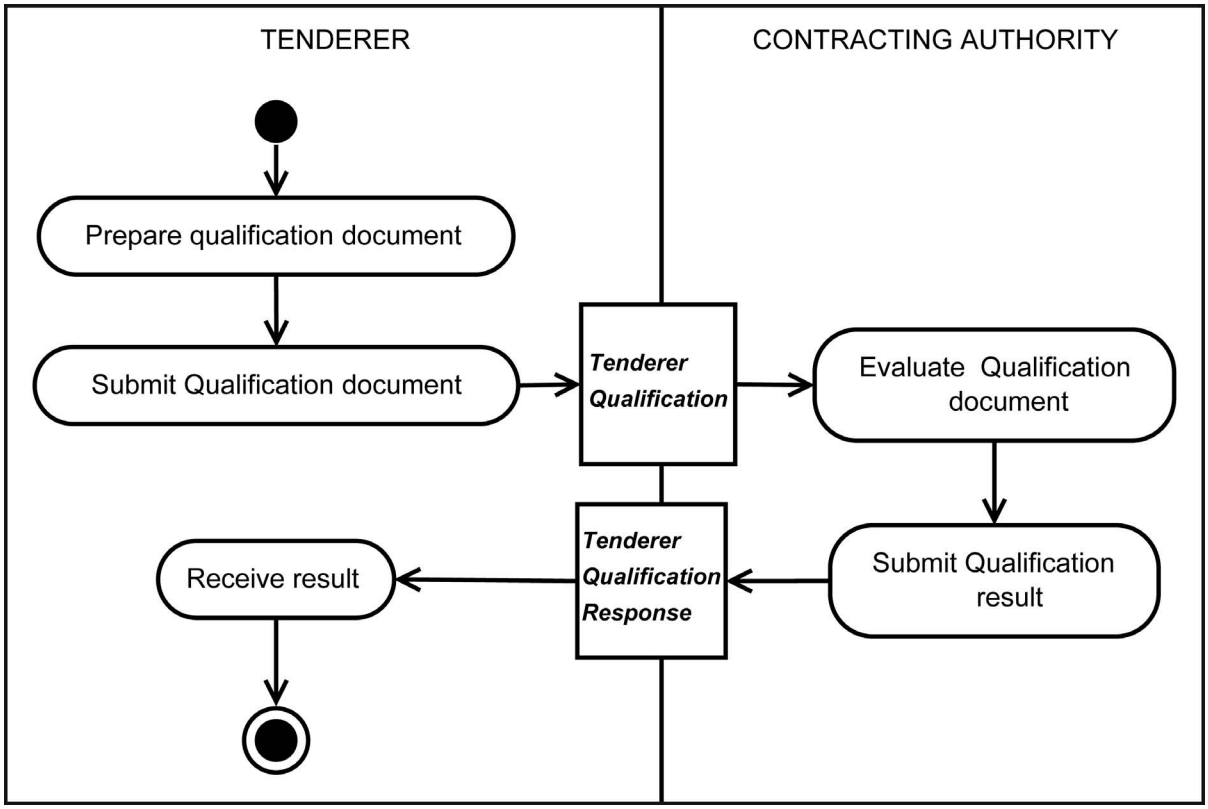
Figure 5. Invitation to Tender



2.3.4 Submission of Qualification Information

The economic operator sends a Tenderer Qualification to the Contracting Authority to *define its own situation or status* relating to the requirements of the Contracting Authority for a specific tendering process. The Contracting Authority uses the Tenderer Qualification Response to notify the Tenderer of its *admission to or exclusion from the tendering process.*

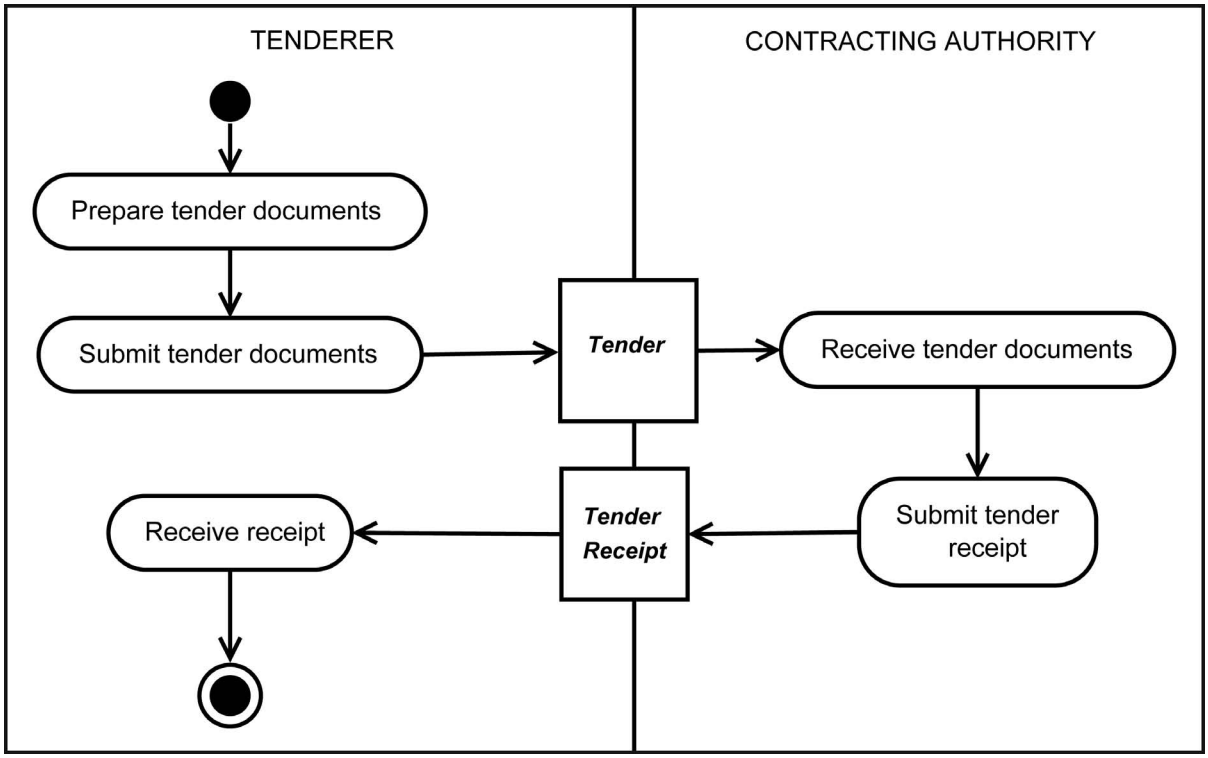
Figure 6. Submission of Qualification Information



2.3.5 Submission of Tenders

A Tenderer submits one or more Tender documents that offer a tender to the Contracting Authority for bid. The Contracting Authority responds with a Tender Receipt to *notify the reception of the tender* for a tendering process. The date and time of the Tender Receipt are significant, because tendering procedures usually have tight deadlines for tender presentation.

Figure 7. Submission of Tenders

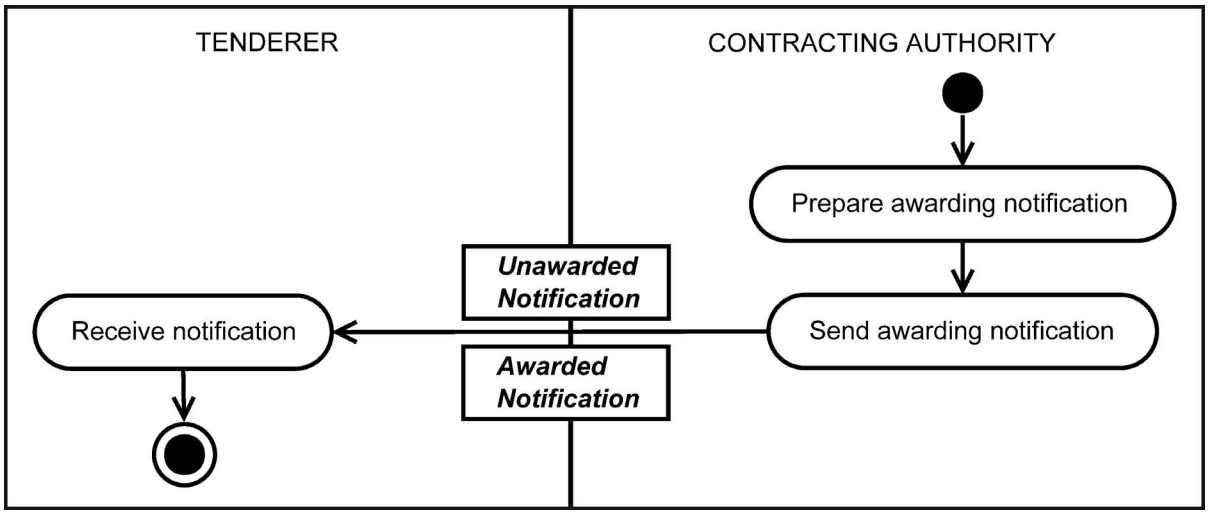


2.3.6 Awarding of Tenders

The awarding of tenders takes place in three phases.

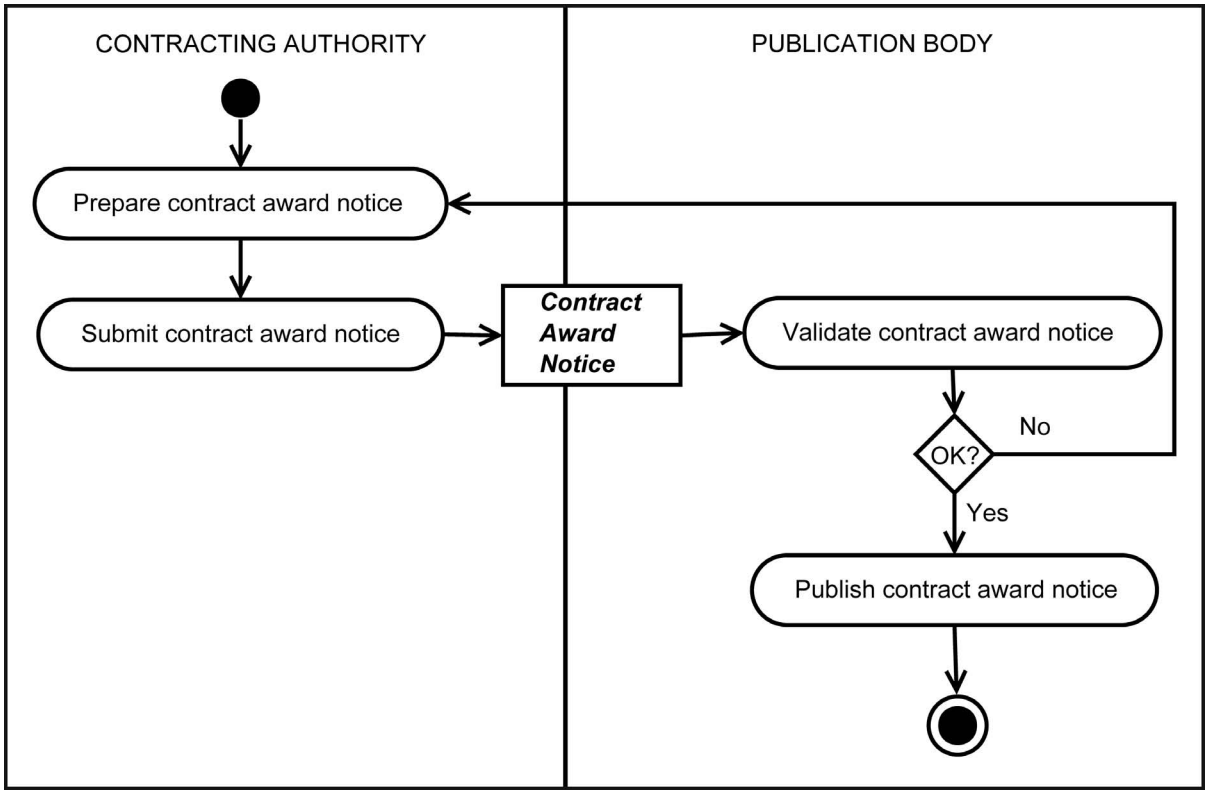
First, the Contracting Authority *notifies each tenderer of its success or failure* in winning the contract, using the Awarded Notification document to communicate the contract award to the winning tenderer or the Unawarded Notification document to communicate that the contract has been awarded to another tenderer.

Figure 8. Award Notification



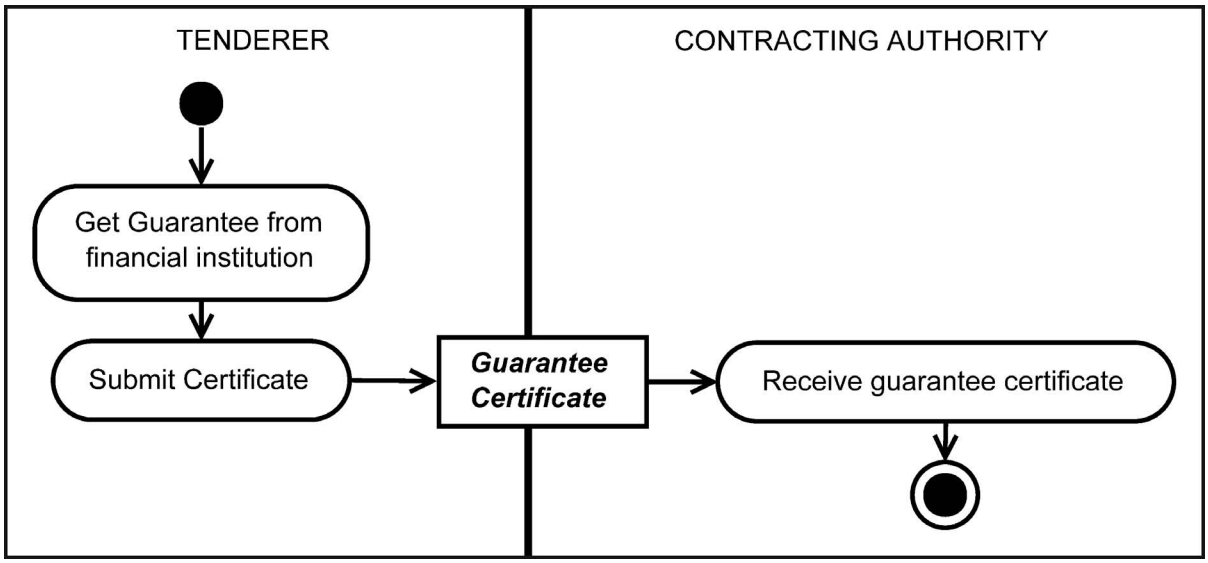
Second, the Contracting Authority causes a Contract Award Notice to be published to *announce the awarding* of a procurement project.

Figure 9. Award Publication



Finally, the Tenderer sends a Guarantee Certificate to *notify the deposit of a guarantee.*

Figure 10. Guarantee Deposit



2.4 Catalogue

A Catalogue is a document produced by a party in the procurement chain that describes items and prices. Document types associated with Catalogue processes are Catalogue Request, Application Response, Catalogue Item Specification Update, Catalogue Pricing Update, and Catalogue Deletion.

2.4.1 Catalogue Business Rules

- Any conditions specified in the contract shall overrule those stated in the common Catalogue.
- A Catalogue exchange shall be between one Provider and one Receiver Party.
- A classification system may have its own set of properties.
- A classification scheme shall have metadata.
- A Catalogue may have a validity period.
- A Catalogue should include item classifications.
- Classification schemes should include standard and specific properties.
- A Catalogue may refer to the lot (sub-section) of a contract.
- A Catalogue may explicitly specify the framework contract reference.
- A Catalogue may refer to a DPS contract number.
- When a Catalogue item is updated, the item shall be replaced in the Catalogue.
- When a Catalogue item is updated, historical information about replaced or updated items must be available to reconcile with outstanding transactions.
- Prices may be updated independently of other Catalogue information.
- Catalogue distribution may be Provider or Receiver Party initiated.
- If a Receiver initiates a request for a Catalogue, they may request an entire Catalogue or only updates to either pricing or item specification details.
- Whether Receiver Party initiated or not, the decision to issue a new Catalogue or update an existing one shall be at the discretion of the Provider Party.
- If an updated Catalogue is issued, then an action code shall define the status of the items in the Catalogue.

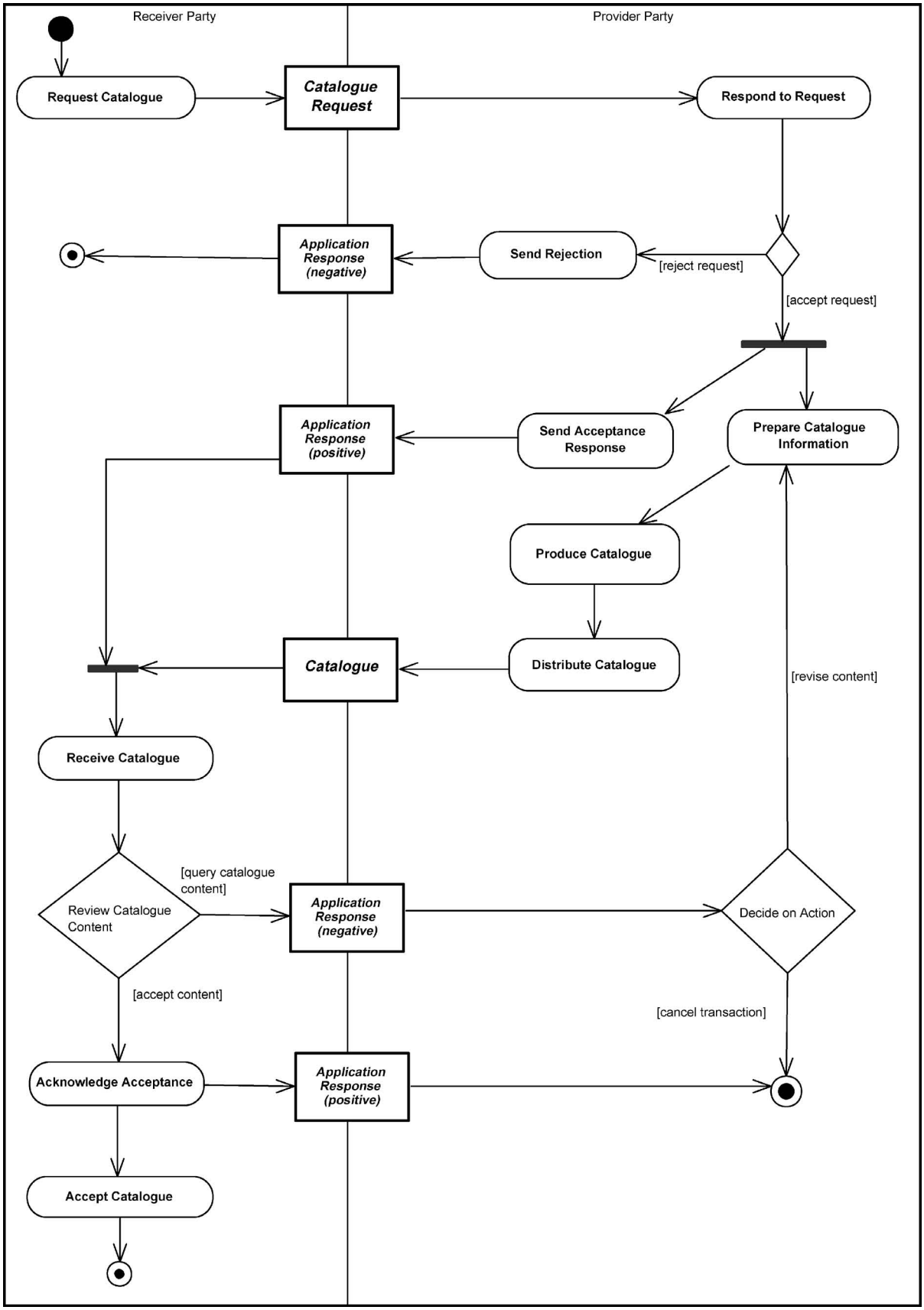
2.4.2 Catalogue Provision

Catalogue provision is the case where a Provider sends information regarding items available for purchase to a Receiver. This may be on request or unsolicited. Because they are only potential purchasers, a Receiver may never become a Customer Party.

2.4.2.1 Create Catalogue

The process of creating a Catalogue is shown in the following diagram. The UBL document types involved are Catalogue, Catalogue Request, and Application Response.

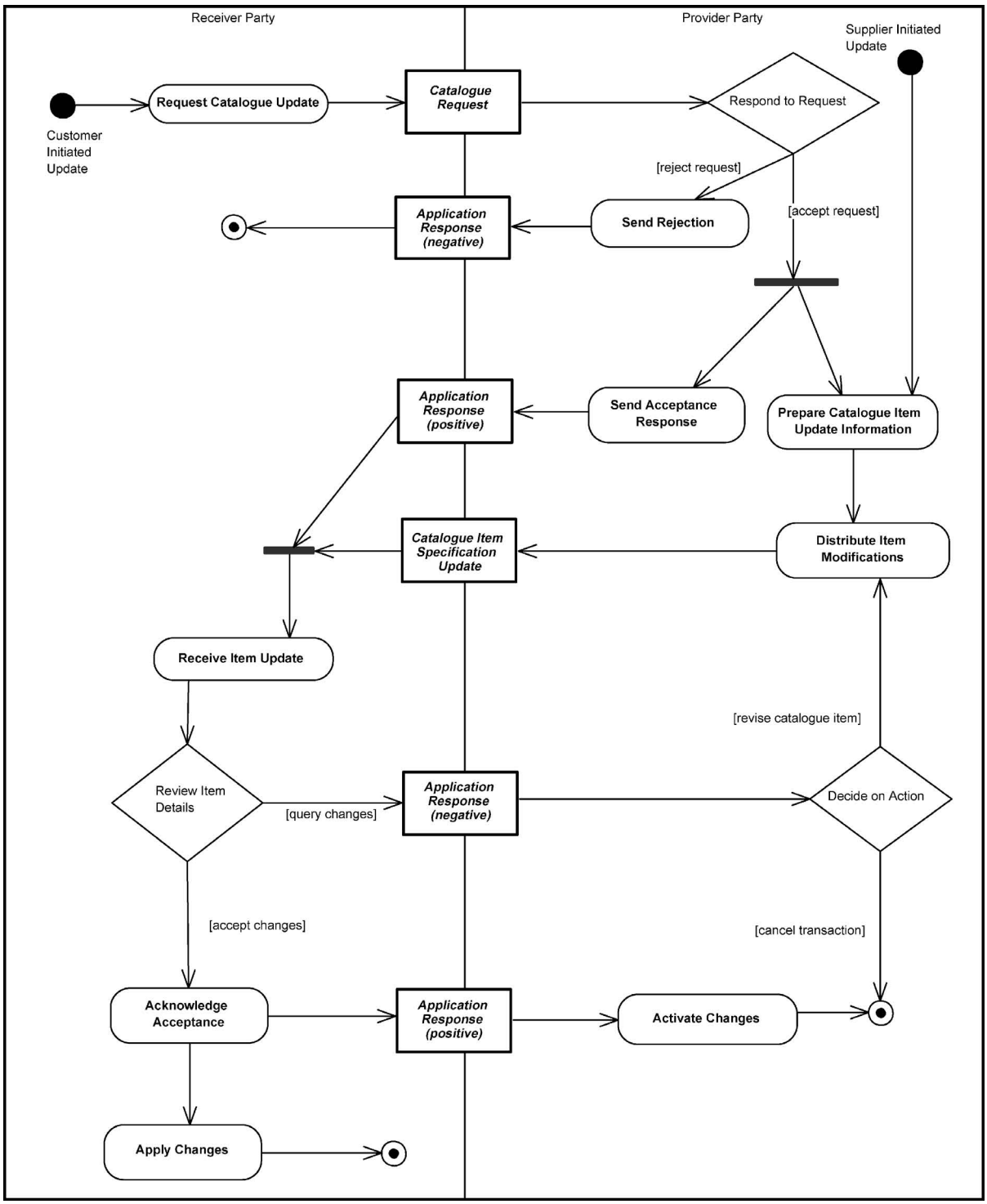
Figure 11. Create Catalogue Process



2.4.2.2 Update Catalogue Item Specification

The process of updating a Catalogue Item specification using Catalogue Item Specification Update is shown in the following diagram. The Catalogue Request and Application Response documents also participate.

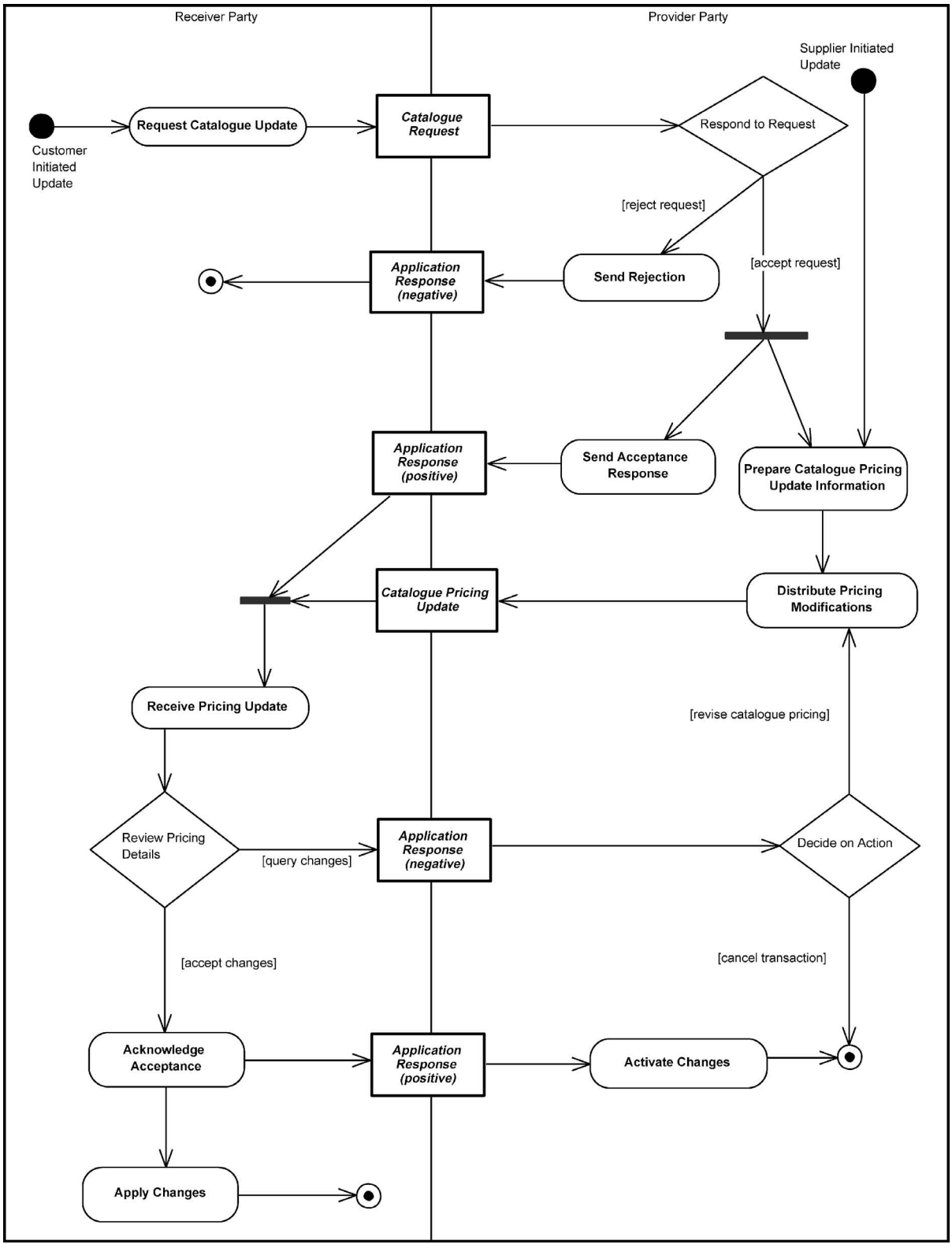
Figure 12. Update Item Specification Process



2.4.2.3 Update Catalogue Pricing

The process of updating Catalogue pricing is shown in the following diagram. The UBL document types involved are Catalogue, Catalogue Request, Catalogue Pricing Update, and Application Response.

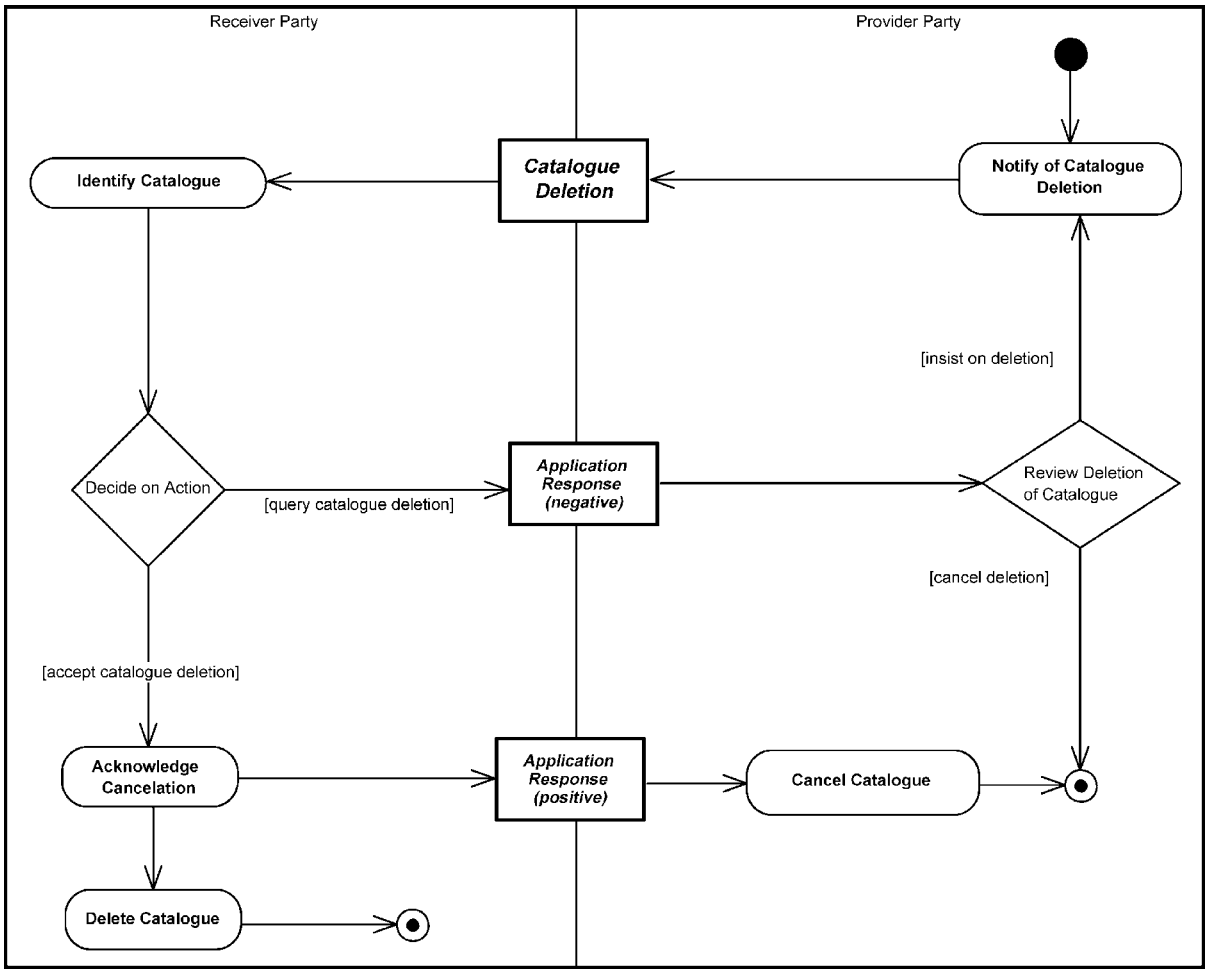
Figure 13. Update Catalogue Pricing Process



2.4.2.4 Delete Catalogue

Deletion of a Catalogue using Catalogue Deletion and Application Response is shown in the following diagram.

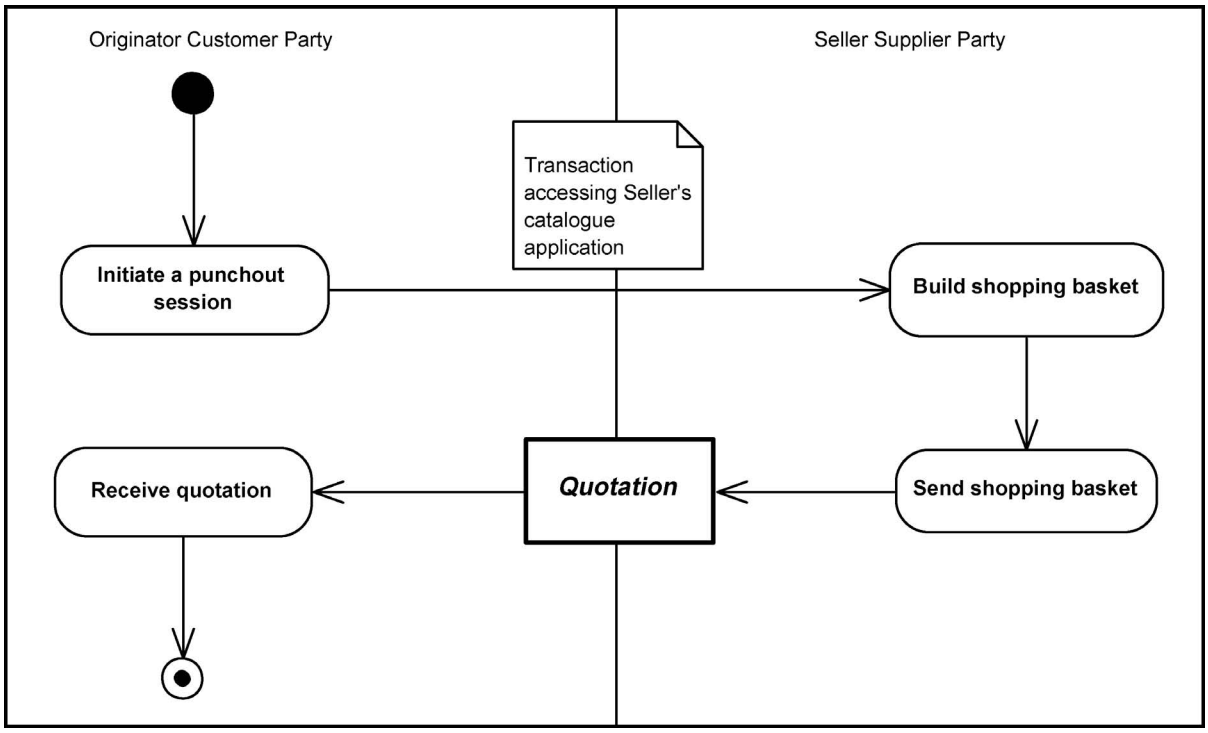
Figure 14. Delete Catalogue Process



2.4.2.5 Punchout

Punchout is a technological innovation whereby an Originator is able to directly access a Seller's application from within the Seller's own procurement application.

Figure 15. Punchout Sourcing Process



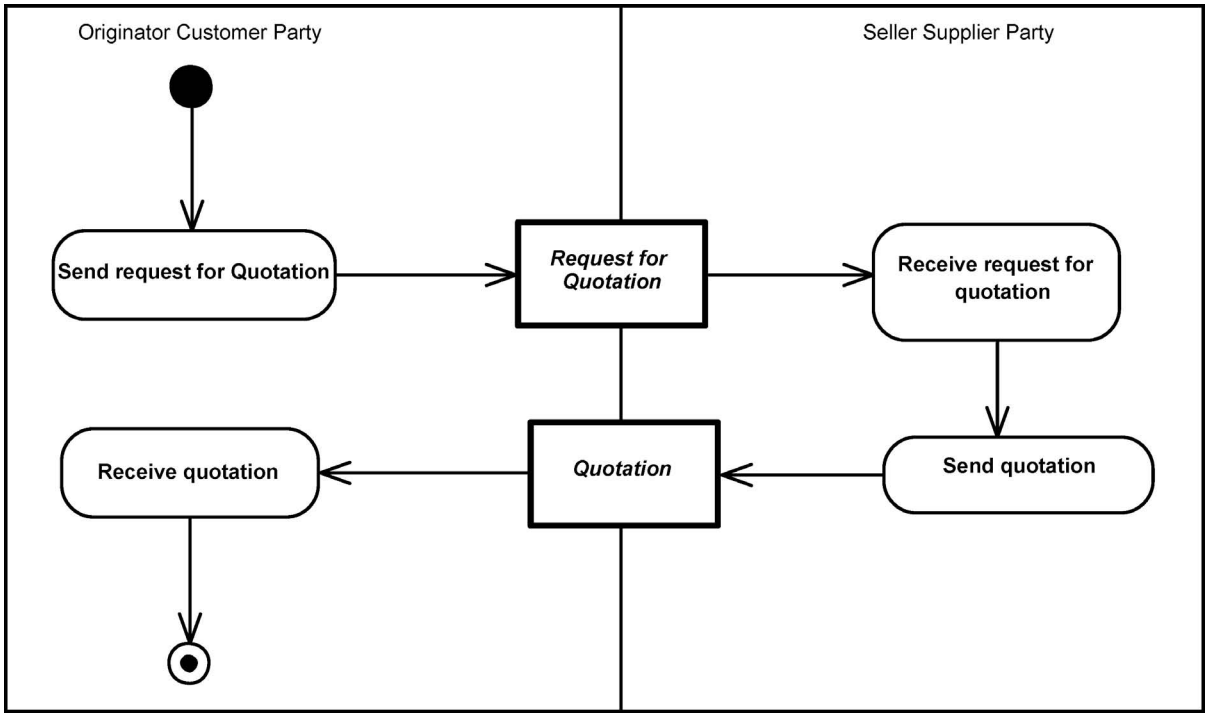
The Originators leave ("punch out" from) their system and interact with the Seller's catalogue to locate and order products, while the Seller's application transparently gathers pertinent information.

While conceptually the punchout request is a form of Request for Quotation (see Section 2.5, "Quotation"), the exchange transaction is tightly coupled to the specific catalogue application and is considered outside the scope of UBL; thus, the only UBL document type involved in this process is Quotation.

2.5 Quotation

Less formally defined than a tender, a quotation process is the case where the Originator asks for a Quotation via a Request for Quotation, as shown in the following diagram.

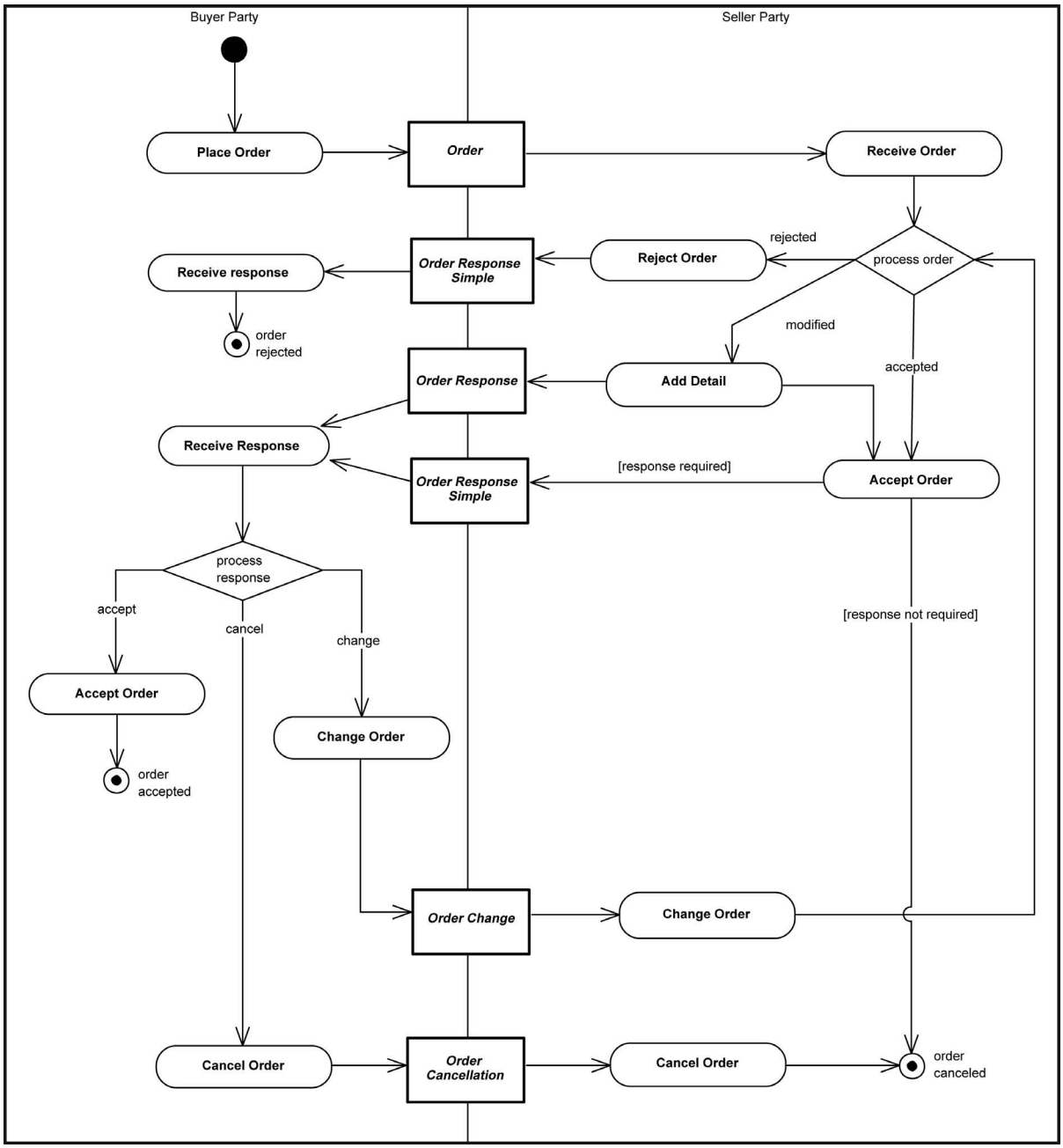
Figure 16. Quotation Process



2.6 Ordering

Ordering is the collaboration that creates a contractual obligation between the Seller Supplier Party and the Buyer Customer Party. Document types in these processes are Order, Order Response, Order Response Simple, Order Change, and Order Cancellation.

Figure 17. Ordering Process



2.6.1 Ordering Business Rules

- The Order may specify allowance and charge instructions (e.g., freight, documentation, etc.) that identify the type of charge and who pays which charges. The Order may be placed “on account” against a trading credit account held by the Seller, or against a credit/debit card account, or against a direct debit agreement. The Order allows for an overall currency defining a default for all pricing and also a specific currency to be used for Invoicing. Within an Order, additional currencies may be specified both for individual item pricing and for any allowances or charges.

- Trade discount may be specified at the Order level. The Buyer may not know the trade discount, in which case it is not specified. This makes a detailed response from the Seller necessary; see Order Response in Section 2.6.3, “Order Response”.
- The Order provides for multiple Order Lines.
- The Order may specify delivery terms, while the Order Line may provide instructions for delivery.
- The Buyer may indicate potential acceptable alternatives.

2.6.2 Order Response Simple

The Order Response Simple is the means by which the Seller confirms receipt of the Order from the Buyer, indicating either commitment to fulfil without change or that the Order has been rejected.

2.6.3 Order Response

Proposed changes to an Order by the Seller are accomplished through the full Order Response document.

The Order Response proposes to replace the original Order. It reflects the entire new state of an order transaction.

It also is the means by which the Seller confirms or supplies Order-related details to the Buyer that were not available to, or specified by, the Buyer at the time of ordering. These may include:

- Delivery date, offered by the Seller if not specifically requested by the Buyer
- Prices
- Discounts
- Charges
- Item Classification codes

The Seller may advise on replacements, substitutes, or other necessary changes using the Order Response.

2.6.4 Order Change

The Buyer may change an established Order in two ways, subject to the legal contract or trading partner agreement: first, by sending an Order Change, or second, by sending an Order Cancellation (see Section 2.6.5, “Order Cancellation”) followed by a new, complete replacement Order.

An Order Change reflects the entire current state of an order transaction.

Buyers may initiate a change to a previously accepted order for various reasons, such as changing ordered items, quantity, delivery date, ship-to address, etc. Suppliers may accept or reject the Order Change using either Order Response or Order Response Simple.

2.6.5 Order Cancellation

At any point of the process, a Buyer may cancel an established order transaction using the Order Cancellation document. Legal contracts, trading partner agreements, and business rules will determine the point at which an Order Cancellation will be ignored (e.g., at the point of manufacture or the initiation of the delivery process). Given the agreements and rules, an Order Cancellation may or may not be an automated business transaction. The terms and conditions of contract formation for business commitments will dictate which, if any, of these restrictions or guidelines will apply.

2.7 Fulfilment

Fulfilment is the collaboration in which the goods or services are transferred from the Despatch Party to the Delivery Party.

Document types in these processes are Despatch Advice, Receipt Advice, Order Cancellation, Order Change, and Fulfilment Cancellation.

In common practice, fulfilment is either supported by a proactive Despatch Advice from the Despatch Party or by a reactive Receipt Advice from the Delivery Party.

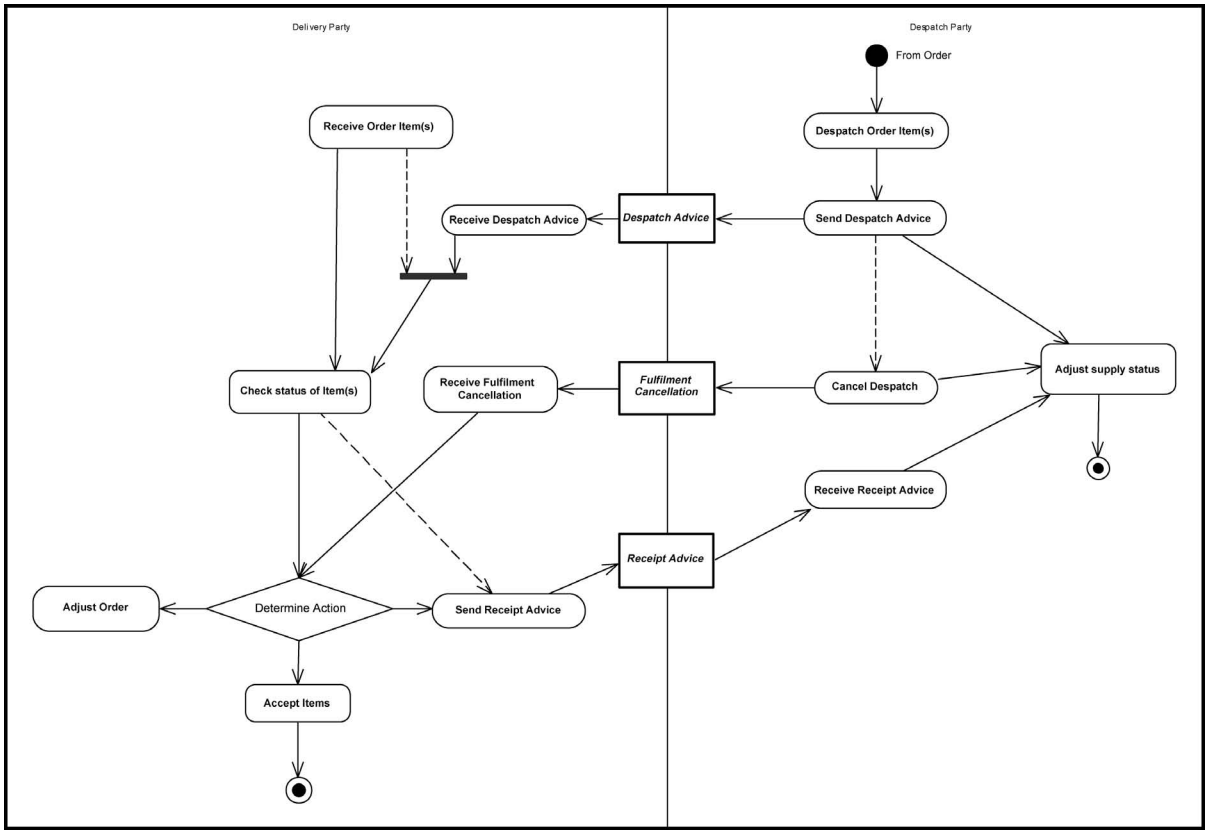
If the Customer is not satisfied with the goods or services, they may then cancel or change the order (see Section 2.6, “Ordering”). The Seller may have a fulfilment (or customer) service dealing with anomalies.

Cancellation of a Despatch Advice or Receipt Advice is accomplished using the Fulfilment Cancellation document (see Section 2.7.3, “Fulfilment Cancellation Business Rules”).

2.7.1 Despatch Advice Business Rules

The Despatch Advice is sent by the Despatch Party to the Delivery Party to confirm shipment of items.

Figure 18. Fulfilment with Despatch Advice Process



The Despatch Advice provides for two situations:

1. Organization of the delivery set of items by Transport Handling Unit(s) so that the Receiver can check the Transport Handling Unit and then the contained items. Quantities of the same item on the same Order Line may be separated into different Transport Handling Units and hence appear on separate Despatch Lines within a Transport Handling Unit.

2. Organization of the delivery set of items by Despatch Line, annotated by the Transport Handling Unit in which they are placed, to facilitate checking against the Order. For convenience, any Order Line split over multiple Transport Handling Units will result in a Despatch Line for each Transport Handling Unit they are contained in.

Additionally, in either case, the Despatch Advice may advise:

- Full Despatch—advising the Recipient and/or Buyer that all the items on the order will be, or are being, delivered in one complete consignment on a given date.
- Partial Despatch—advising the Recipient and/or Buyer that the items on the order will be, or are being, partially delivered in a consignment on a given date.

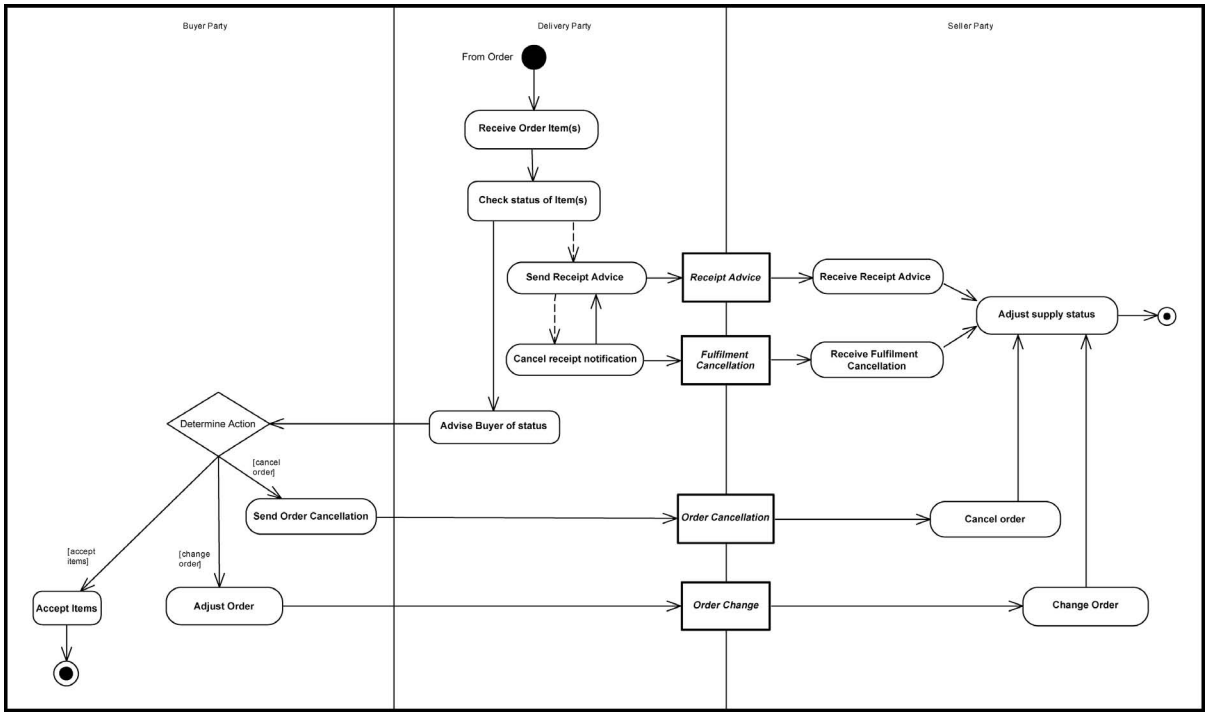
Despatch Lines of the Despatch Advice need not correspond one-to-one with Order Lines, and are linked by a reference. The information structure of the Despatch Advice may result in multiple Despatch Lines from one Order Line. Equally, partial despatch may result in some Order Lines not being matched by any Line in a Despatch Advice.

Within a Despatch Advice, an Item may also indicate the Country of Origin and the Hazardous nature of the Item.

2.7.2 Receipt Advice Business Rules

The Receipt Advice is sent by the Delivery Party to the Despatch Party to confirm receipt of items and is capable of reporting shortages or damaged items.

Figure 19. Fulfilment with Receipt Advice Process



The Receipt Advice provides for two situations. For ease of processing claimed receipt against claimed delivery, it must be organised in the same way as the corresponding Despatch Advice:

1. Indication of receipt by Transport Handling Unit(s) and contained Receipt Lines one-to-one with the Despatch Advice as detailed by the Seller party, or

2. Indication of receipt by Receipt Lines annotated by Transport Handling Unit, one-to-one with the Despatch Advice as detailed by the Seller party.

The Receipt Advice allows the Delivery Party to state any shortages from the claimed despatch quantity and to state any quantities rejected for a given reason.

2.7.3 Fulfilment Cancellation Business Rules

In real life, the sender of a Despatch Advice or Receipt Advice sometimes needs to cancel the document after it has been sent. The Fulfilment Cancellation document is provided for this purpose.

For example, a Despatch Advice may later be cancelled by the Supplier when a problem with shipment prevents the delivery of goods, or the goods to be shipped are not available, or the order is cancelled; in these cases, the customer cancels receipt and adjusts the order accordingly (see Figure 18, "Fulfilment with Despatch Advice Process").

Similarly, a Receipt Advice may later be cancelled by the customer (see Figure 19, "Fulfilment with Receipt Advice Process") if the customer discovers an error in ordering (failure to follow formal contractual obligations, incorrect product identification, etc.) or a problem with a delivered item (malfunction, missing part, etc.). In this case, the billing and payment process may be put on hold.

2.8 Billing

In the Billing process, a request is made for payment for goods or services that have been ordered, received, or consumed. In practice, there are several ways in which goods or services may be billed.

Document types in these processes are Invoice, Credit Note, Debit Note, and Application Response.

For UBL 2.1, we assume the following billing methods:

1. Traditional Billing
    a. Using Credit Note
    b. Using Debit Note
2. Self Billing (also known as billing on receipt)
    a. Using Credit Note
    b. Using Self Billed Credit Note

2.8.1 Billing Business Rules

An Invoice defines the financial consequences of a business transaction. The Invoice is normally issued on the basis of one despatch event triggering one Invoice. An Invoice may also be issued for pre-payment on a whole or partial basis. The possibilities are:

- Prepayment invoice (payment expected)
- Pro-forma invoice (pre advice, payment not expected)
- Normal Invoice, on despatch for despatched items
- Invoice after return of Receipt Advice

The Invoice only contains the information that is necessary for invoicing purposes. It does not reiterate any information already established in the Order, Order Change, Order Response, Despatch Advice, or Receipt Advice that is not necessary when invoicing. If necessary, the Invoice refers to the Order, Despatch Advice, or Receipt Advice by a Reference for those documents.

Taxation on the Invoice allows for compound taxes, the sequence of calculation being implied by the sequence of information repeated in the data stream (e.g., Energy tax, with VAT—Value Added Tax—superimposed).

Charges may be specified either as a lump sum or by percentage applied to the whole Invoice value prior to calculation of taxes. Such charges cover:

- Packaging
- Delivery/postage
- Freight
- Documentation

Each Invoice Line refers to any related Order Line(s) and may also refer to the Despatch Line and/or Receipt Line.

2.8.2 Traditional Billing

Traditional billing is where the supplier invoices the customer when the goods are delivered or the services provided. In this case, the invoice may be created at the time of despatch or when the Delivery Party acknowledges that the goods have been received (using a Receipt Advice).

When there are discrepancies between the Despatch Advice, Receipt Advice, or Invoice and the goods actually received, or the goods are rejected for quality reasons, the customer may send an Application Response or a Debit Note to the supplier. The supplier may then issue a Credit Note or another Invoice as required.

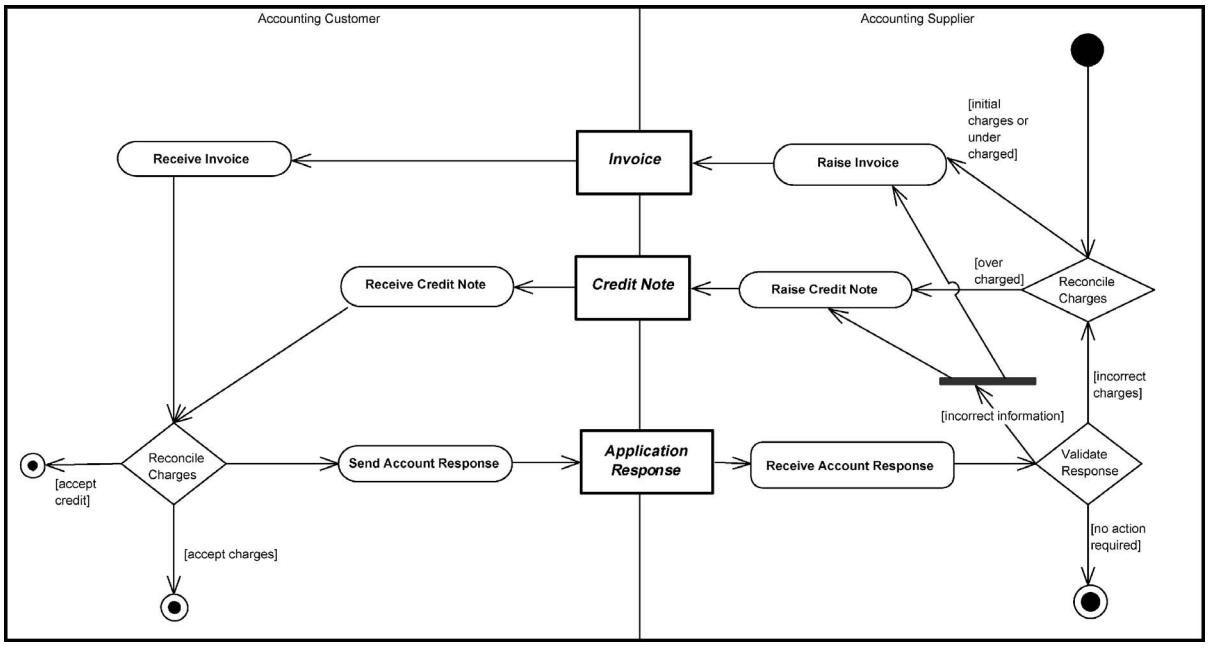
A Credit Note or Debit Note may also be issued in the case of retrospective price change.

Credit Notes or Debit Notes may be also issued after the Billing collaboration (as part of the Payment collaboration).

2.8.2.1 Billing Using Credit Notes

Billing using Credit Note is shown in the following diagram.

Figure 20. Billing with Credit Note Process

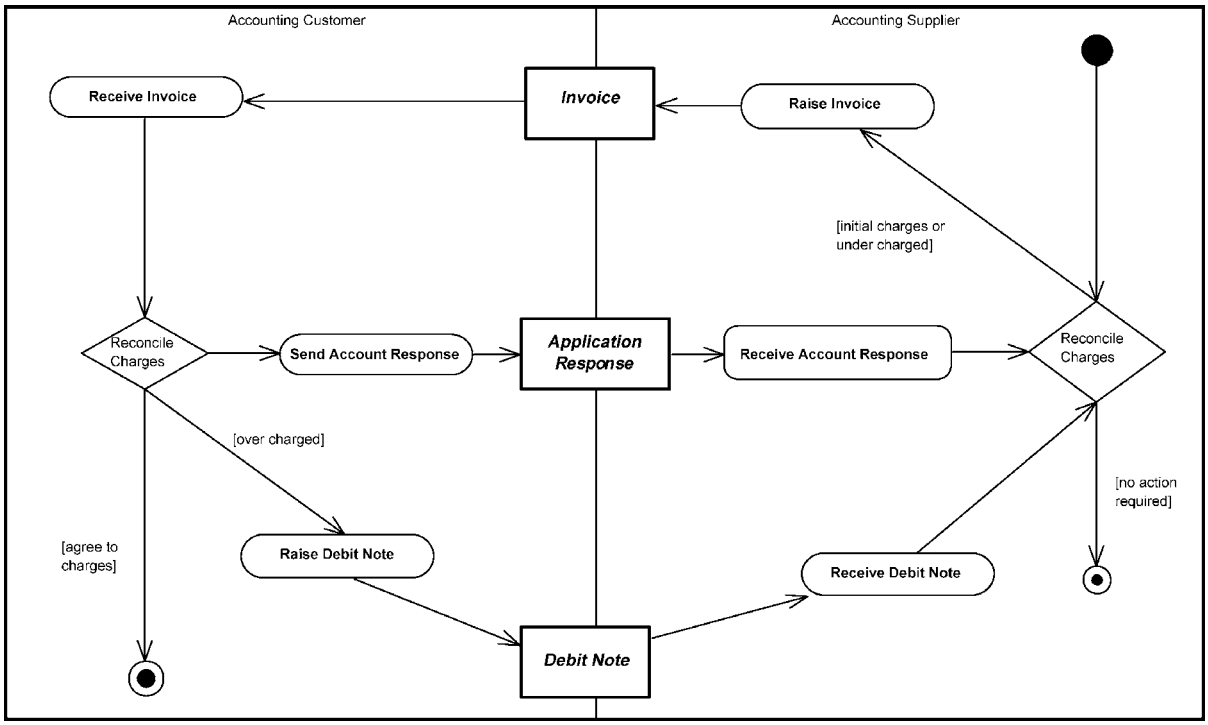


When using Credit Notes, the Supplier (in their Accounting role) is responsible for specifying the tax requirements.

2.8.2.2 Billing Using Debit Notes

Billing using Debit Note is shown in the following diagram.

Figure 21. Billing with Debit Note Process



When using Debit Notes, both the Supplier (in their Accounting role) and the Customer (in their Accounting role) are responsible for providing taxation information.

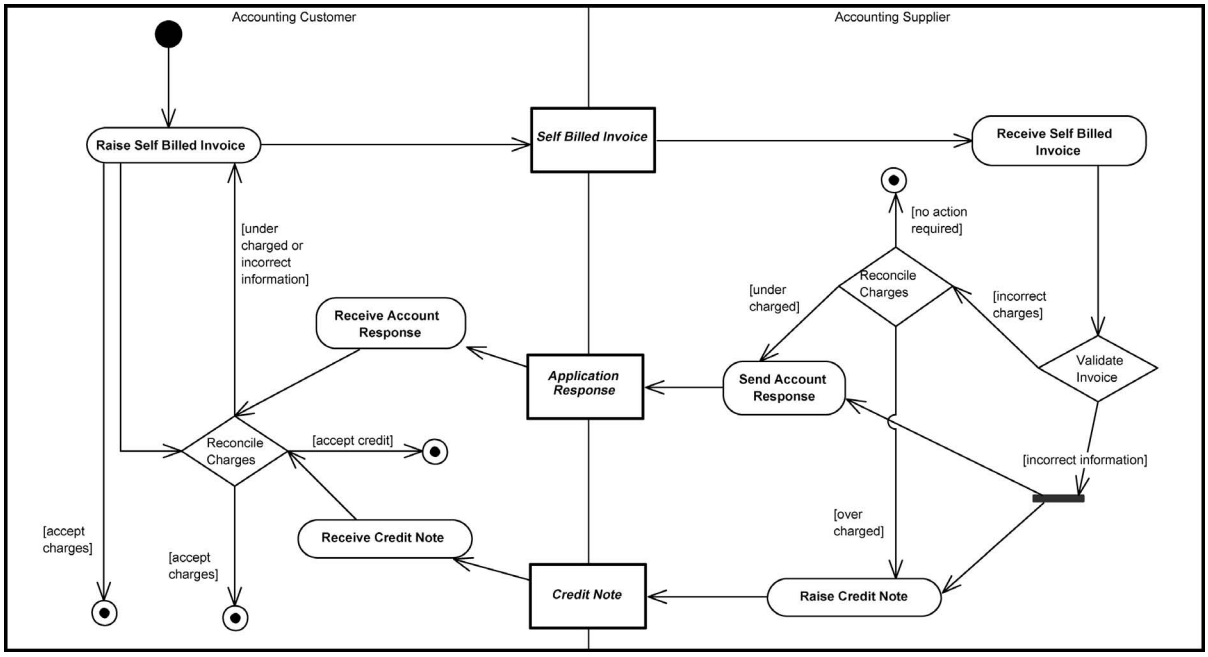
2.8.3 Self Billing

A self billing process is where a Customer “invoices” itself, *in the name and on behalf of* the Supplier, and provides the Supplier with a copy of the self billed invoice.

2.8.3.1 Self Billing Using Credit Notes

Self Billing using Credit Note is shown in the following diagram.

Figure 22. Self Billing with Credit Note Process

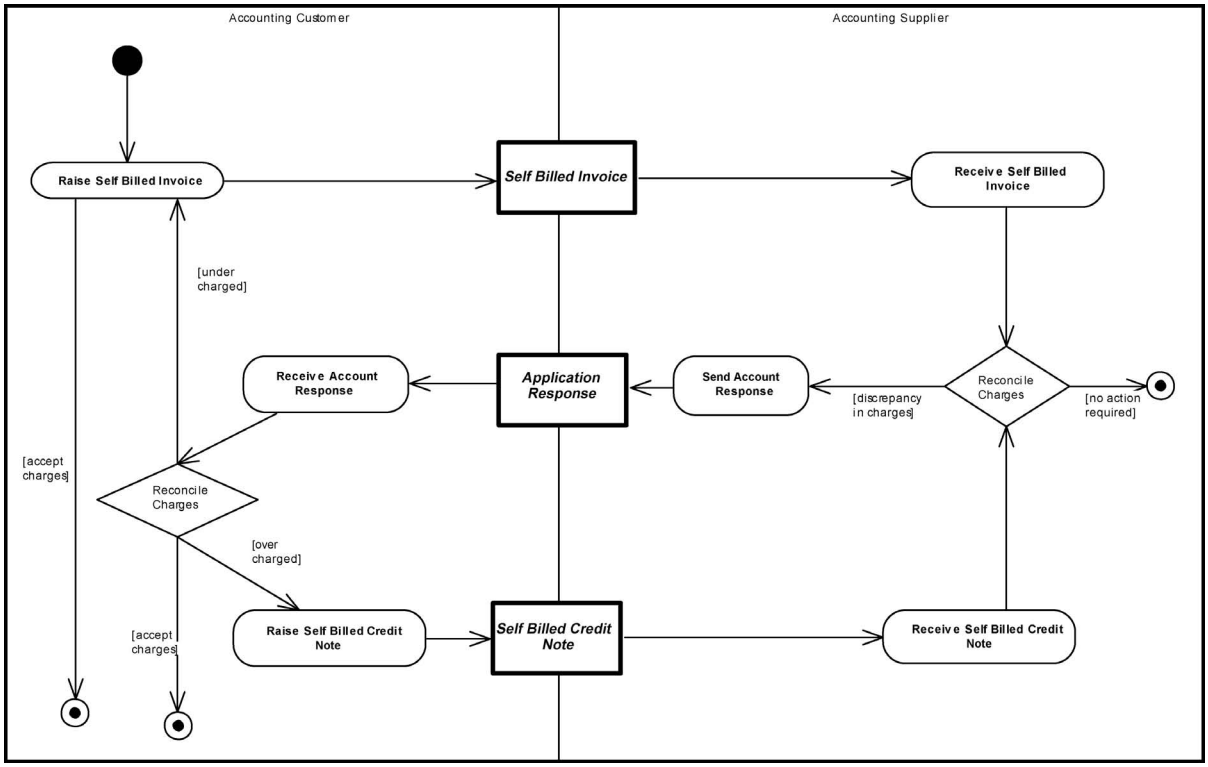


If the Supplier finds that the Self Billed Invoice is incorrect, e.g., wrong quantities or wrong prices, or if the goods have not been invoiced at all, it may send an Application Response or a Credit Note to the Customer. The customer may then verify whether the adjustment is acceptable or not and consequently issue another Self Billed Invoice or a Self Billed Credit Note.

2.8.3.2 Self Billing Using Self Billed Credit Notes

Self Billing using Self Billed Credit Note is shown in the following diagram.

Figure 23. Self Billing with Self Billed Credit Note Process

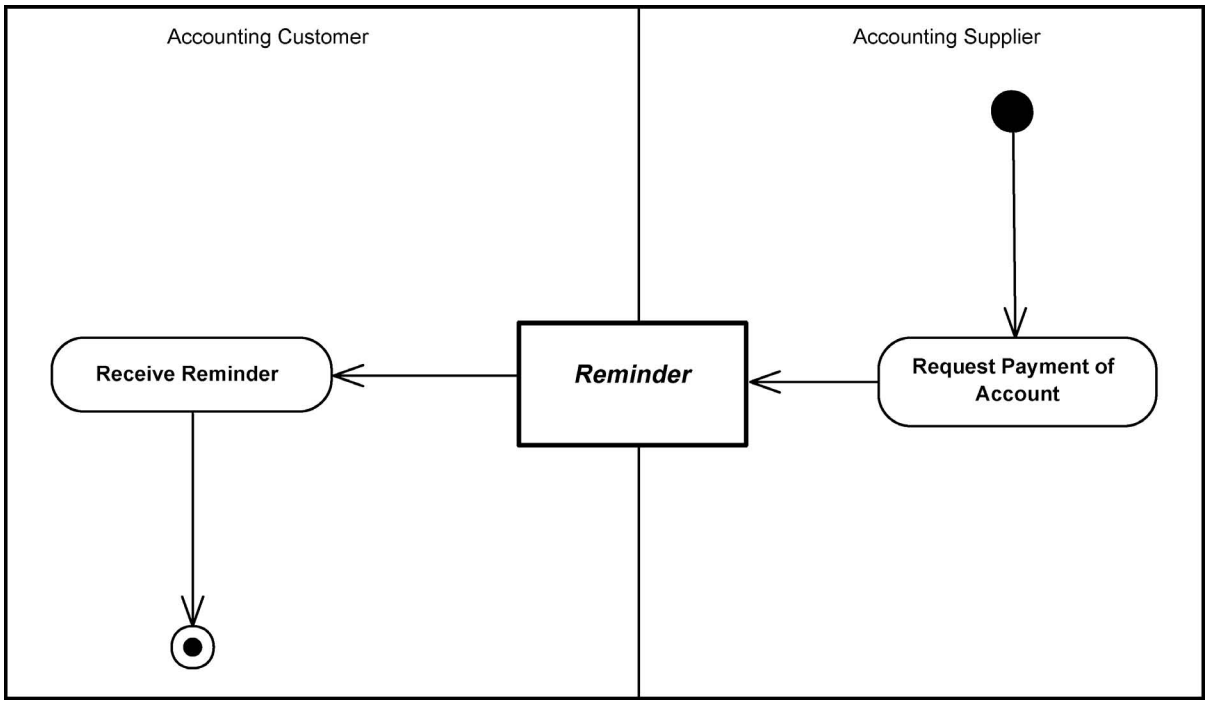


When using Self Billed Credit Notes, the Customer is raising the Self Billed Credit Note *in the name and on behalf of* the Supplier. Therefore the Supplier and the Customer are still both responsible for providing taxation information.

2.8.4 Reminder for Payment

A Reminder may be used to notify the Customer of accounts due to be paid.

Figure 24. Reminder for Payment Process



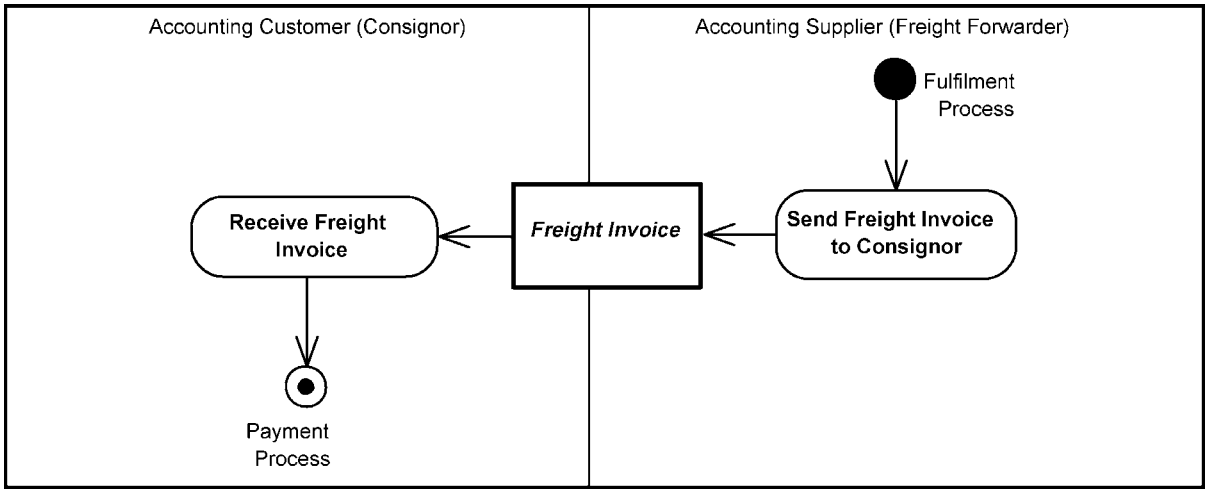
2.9 Freight Billing

An extension of the Billing process is that of Freight Billing. This represents the billing process between the Transport Service Buyer and Transport Service Provider through the use of an invoice for freight charges.

The Transport Service Provider initiates the process of billing the Transport Service Buyer for logistic services.

The Freight Invoice lists the charges incurred in order to fulfil the agreed service.

Figure 25. Freight Billing Process

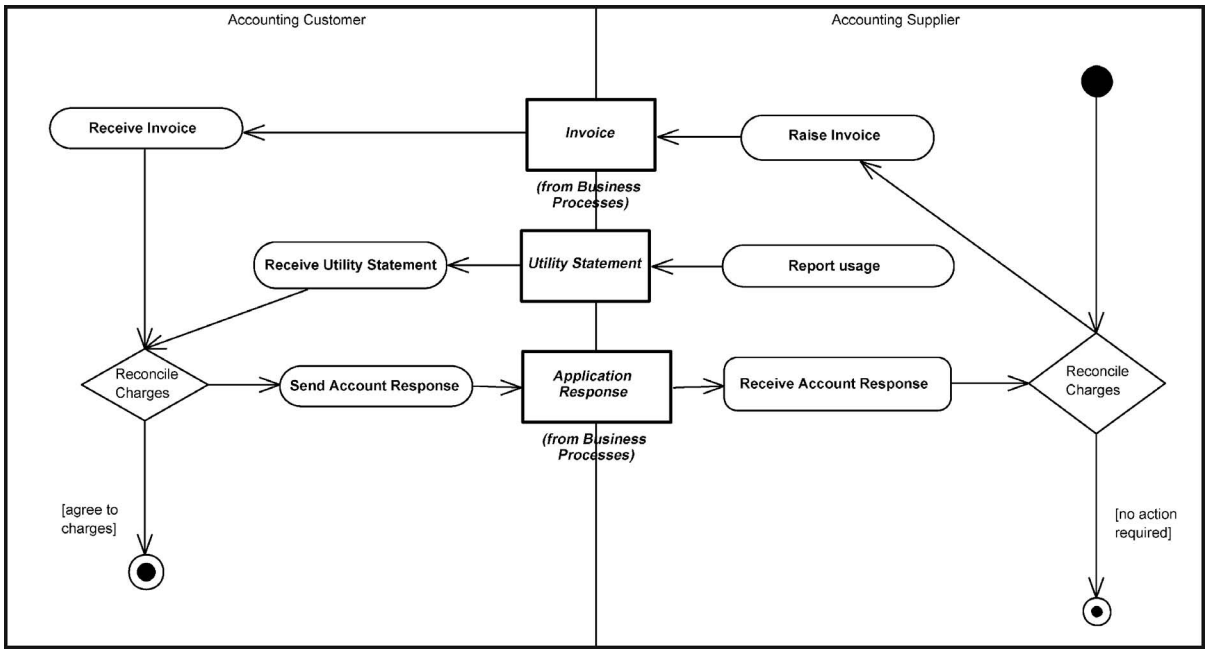


2.10 Utility Billing

This process defines the billing process for invoicing between suppliers of utilities (including electricity, gas, water, and telephony services) and private and public customers.

The Utility Statement supplements an Invoice with information about consumption of the utility's services. An invoice may refer to one or more utility statements, and a utility statement may refer to one or more invoices.

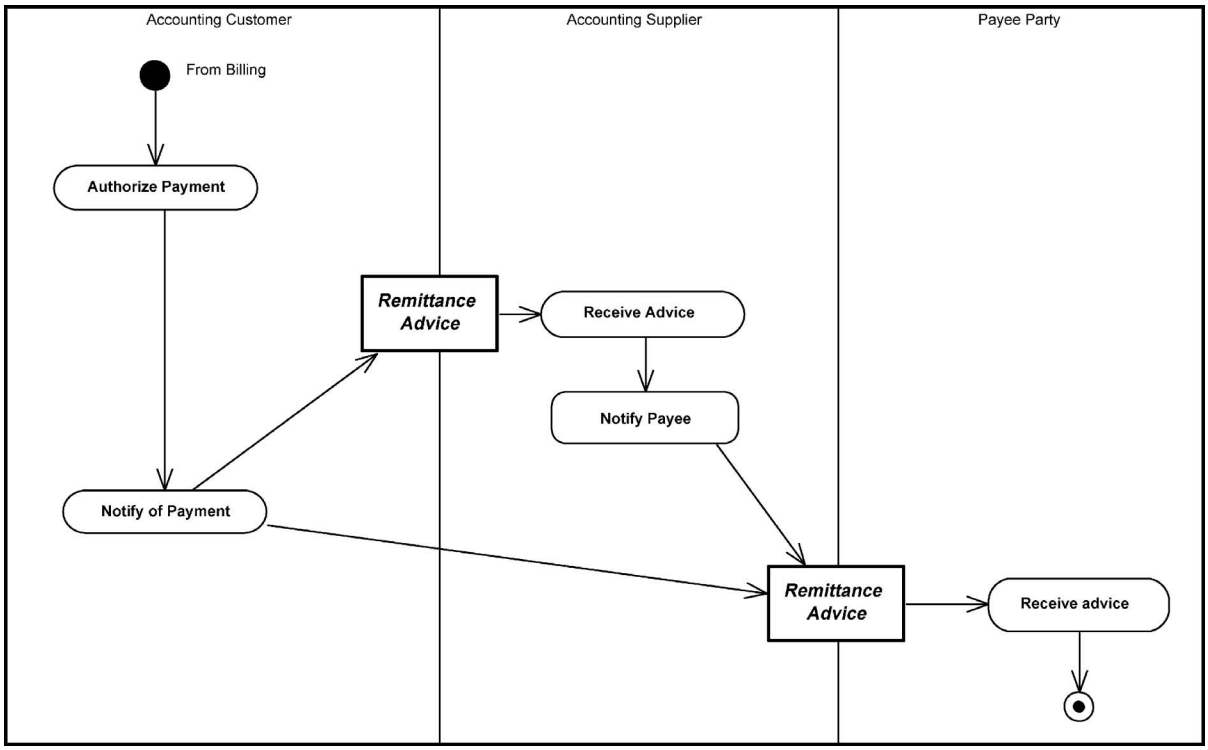
Figure 26. Utility Billing Process



2.11 Payment Notification

In the payment notification process, the Payee (who is most often the Accounting Customer) is notified of any funds transferred, against the account of the Accounting Supplier, using a Remittance Advice document.

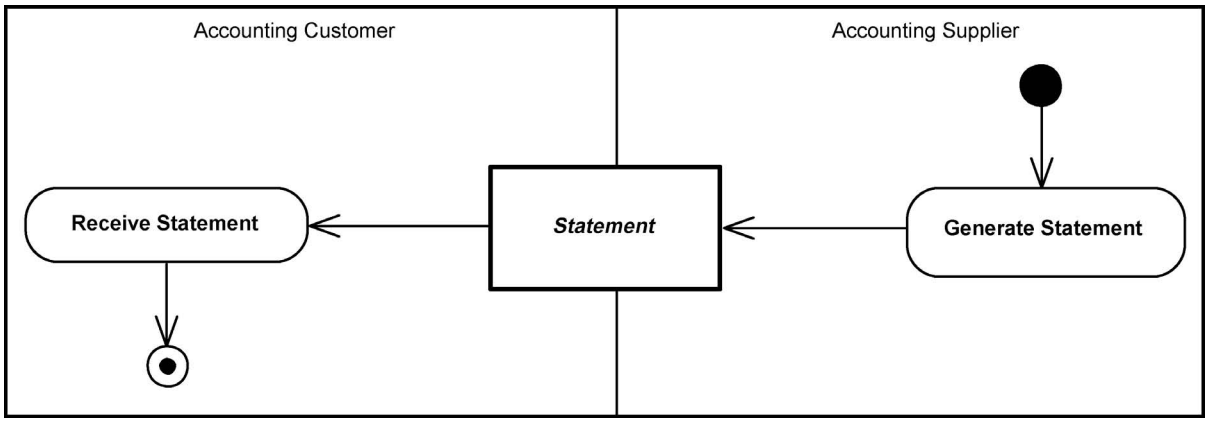
Figure 27. Payment Notification Process



2.12 Report State of Accounts

A Statement of account may be used to notify the Accounting Customer of the status of the billing.

Figure 28. Statement Process



2.13 Collaborative Planning, Forecasting, and Replenishment

The VICS Collaborative Planning, Forecasting, and Replenishment (CPFR®) guidelines [**CPFR**] formalize the processes by which two trading partners agree upon a joint plan to forecast and monitor sales through replenishment and to recognize and respond to any exceptions.

In the UBL 2.1 context of use, these CPFR processes between the retailer and the manufacturer have been extended to cover the planning process between other parties such as the manufacturer and the supplier. These binary collaboration definitions are the template guidelines for implementers to build their own collaboration process based on their supply chain topology and requirements.

As shown in Figure 2-2 of [**CPFR**], the seller and the buyer engage in three main activities in order to improve the overall performance of the supply chain:

1. **Planning** establishes the ground rules for the collaborative relationship. Trading partners exchange information about their corporate strategies and business plans in order to collaborate in the development of a Joint Business Plan. The Joint Business Plan identifies the significant events that affect supply and demand in the planning period, such as promotions, inventory policy changes, store openings/closings, and product introductions.

2. The **Forecasting** phase involves the development of a shared plan based on consumer demand. Estimation of consumer demand at the point of sale is called sales forecasting, and future product ordering based on the sales forecast is referred to as order forecast.

3. The **Replenishment** phase involves order generation, which transitions forecasts to firm demand, and order fulfilment, the process of producing, shipping, delivering, and stocking products for consumer purchase. Note: This phase may be implemented using other UBL processes.

A fourth collaborative activity, **Analysis,** involves monitoring the execution of activities for exceptions that are identified during the strategy and planning phase. Calculation of key performance metrics and plan adjustments for improving results also take place in Analysis. This activity is represented in the CPFR diagram by the arrows labeled "Exception Triggers" and the process called "Resolve/Collaborate on Exception Items" in the Forecasting phase.

While these collaboration activities are presented in logical order, most companies are involved in all of them at any moment in time. There is no predefined sequence of steps. Execution issues can impact strategy, and analysis can lead to adjustments in forecasts.

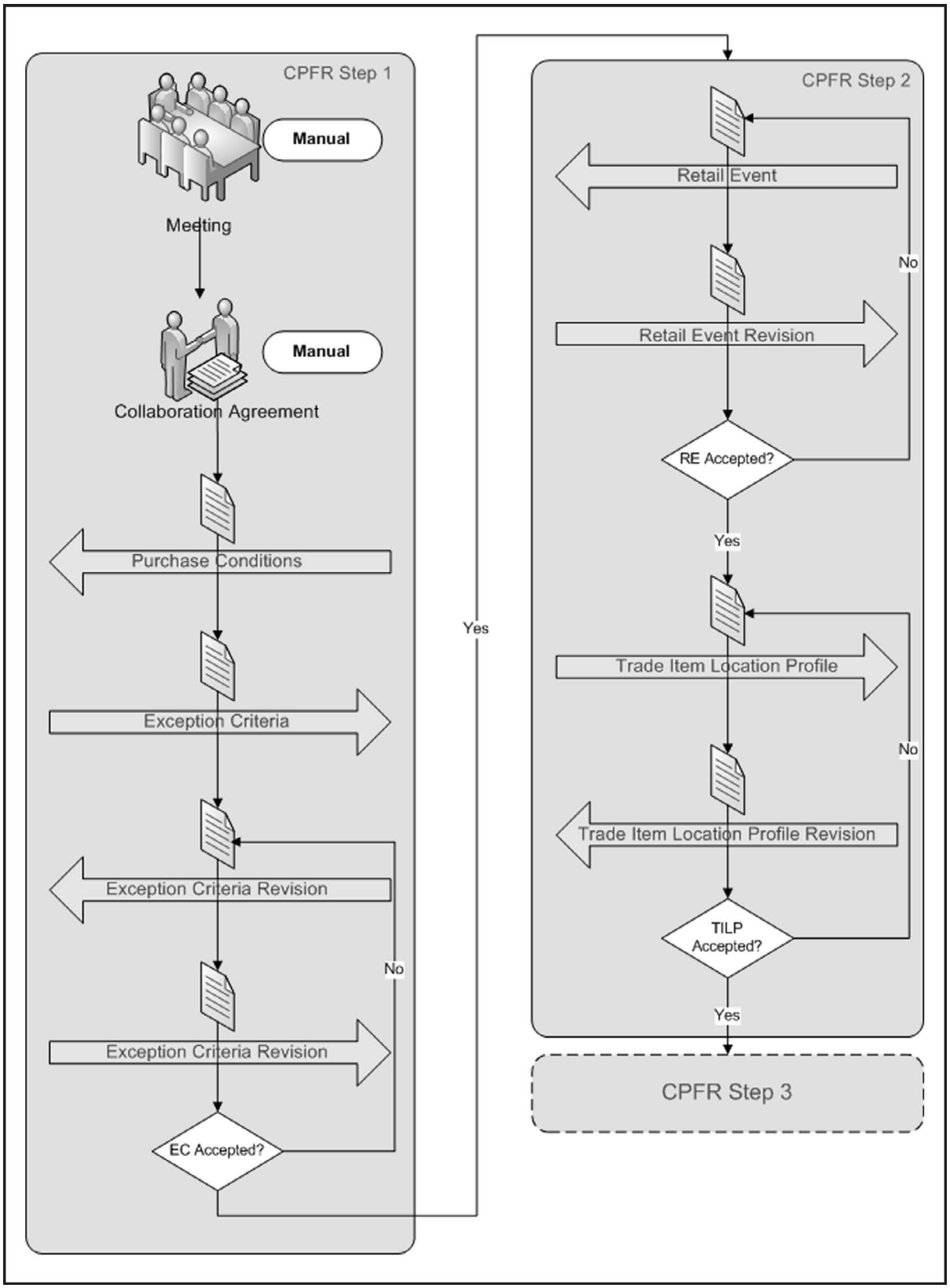
2.13.1 Collaboration Agreement and Joint Business Planning

The Collaboration Arrangement is the preparatory step that defines the scope of the project, assigns roles, establishes procedures for data interchange, and issues identification and resolution. The following actions are performed through meetings and agreements:

- Receive and review background information from the sales organization or buyers
- Identify the product categories that should be included in the initial scope
- Define Collaboration Objectives
- Define specific metrics that reflect the objectives
- Determine the Event collaboration cycle
- Determine the times of the review meetings to discuss the results
- Document the data sources that are essential for a successful event collaboration process, and

- Document additional information that can be used in the event analysis.

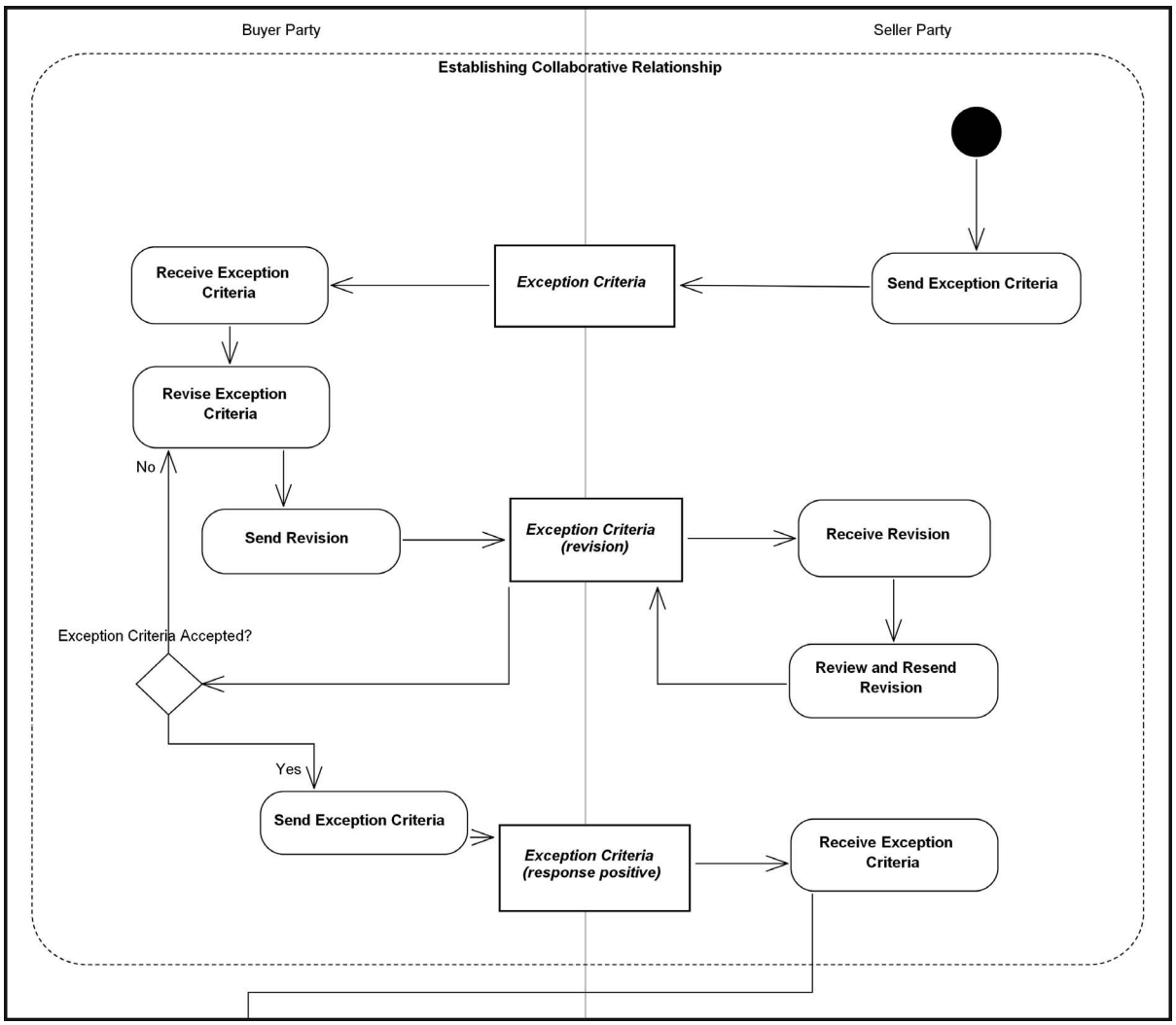
Figure 29. CPFR Steps 1 and 2



The first step of the CPFR Process continues with the exchange of messages containing purchase conditions. (UBL 2.1 does not standardize the format of such messages.) Afterwards, for determining

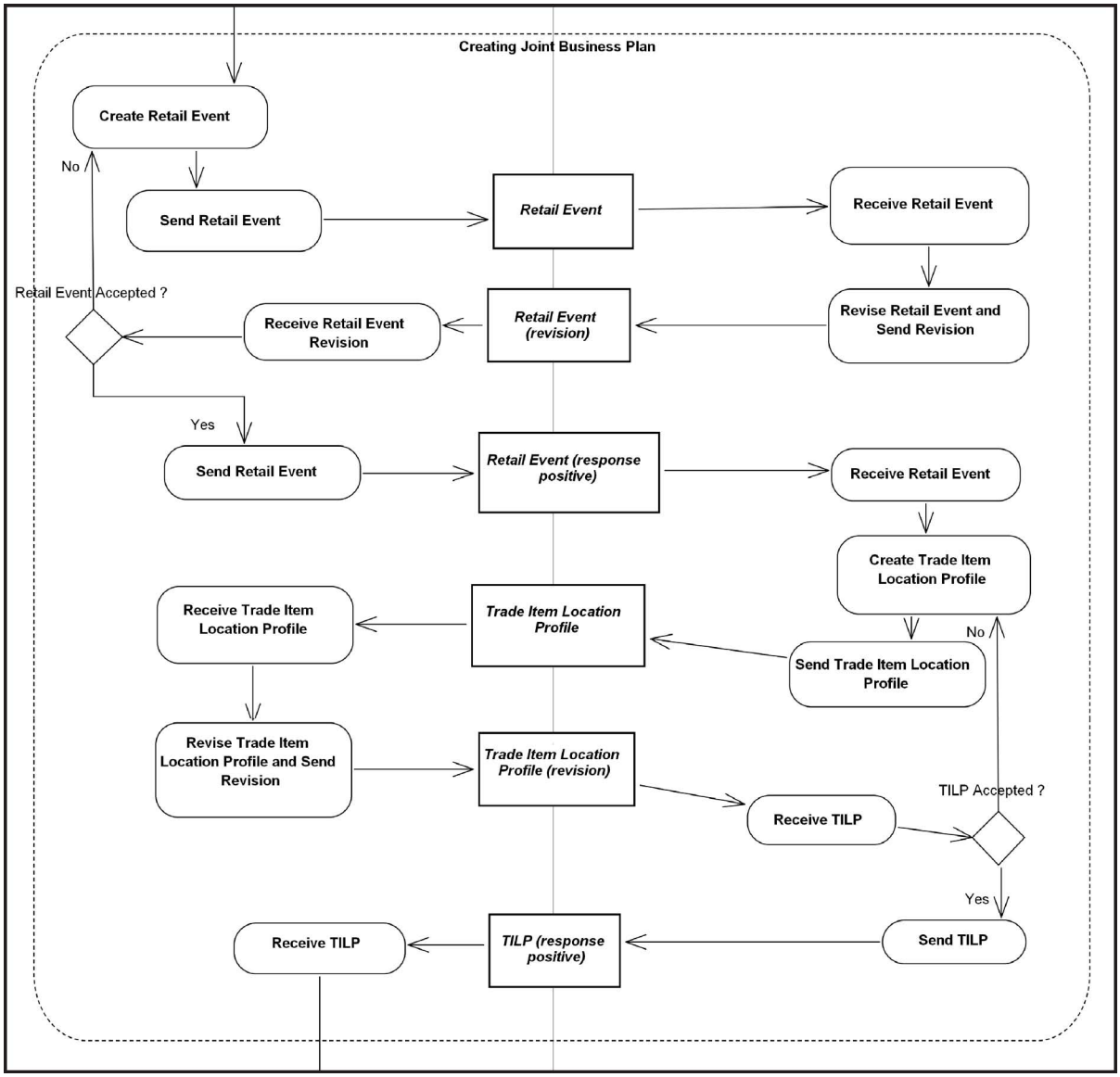
the exception criteria that should be monitored and handled during the execution, Exception Criteria messages are exchanged. Exchange of revised Exception Criteria messages continues until the criteria are accepted by both sides.

Figure 30. Establish Collaborative Relationships



In CPFR Step 2 (the Joint Business Planning phase) there are two messages that should be exchanged and agreed upon: Retail Event and Trade Item Location Profile. Revisions are exchanged until an agreement is achieved.

Figure 31. Create Joint Business Plan

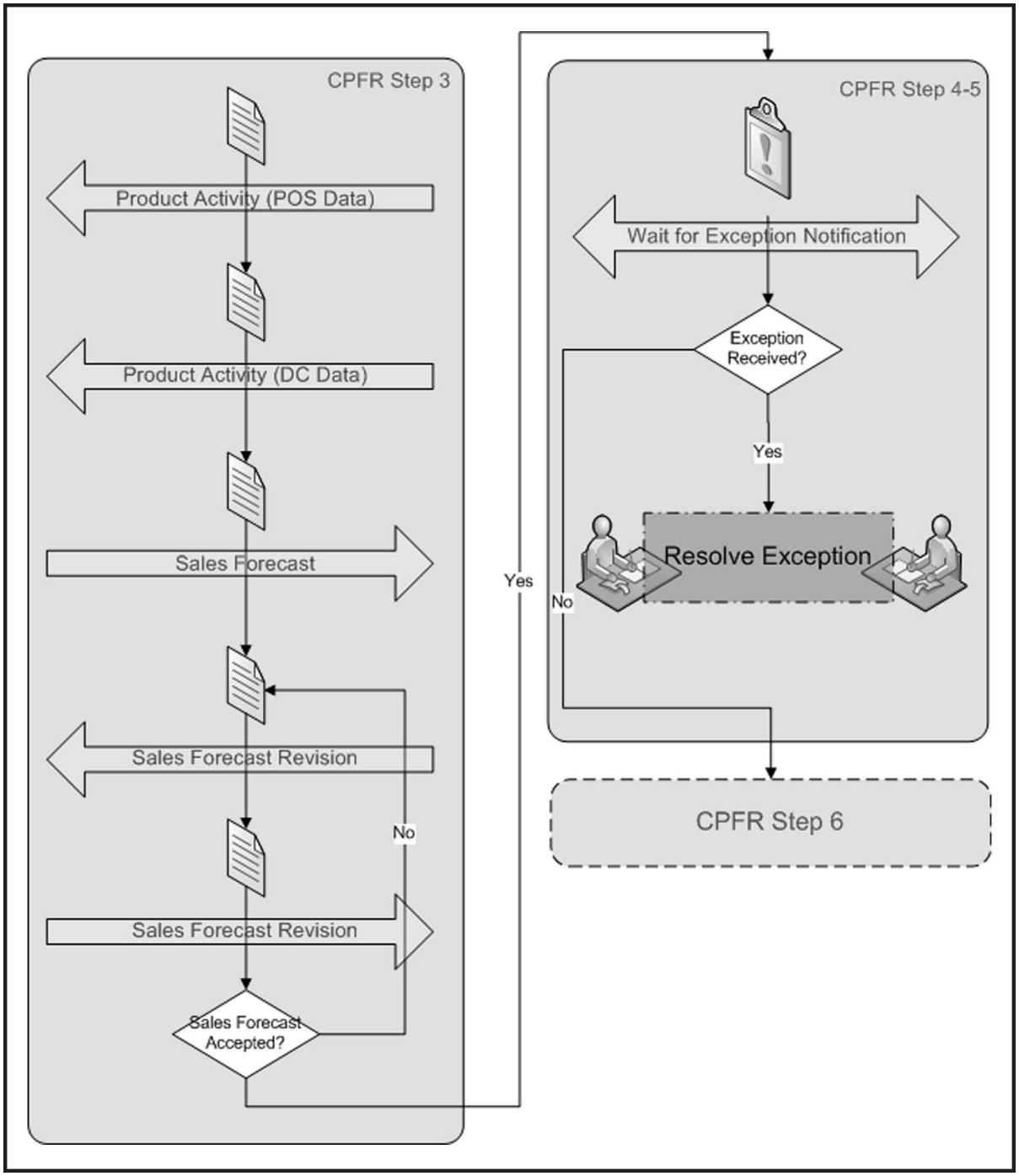


2.13.2 Sales Forecast Generation and Exception Handling

CPFR Step 2 helps the buyer and seller agree to the event details and calendar that meet their joint business and collaboration objectives. The objective of the event calendar is to ensure that events are planned to achieve the optimal results and to enable both parties to plan the execution of the event more accurately, from the preparation of advertising and displays to the production and delivery of the promotional stock.

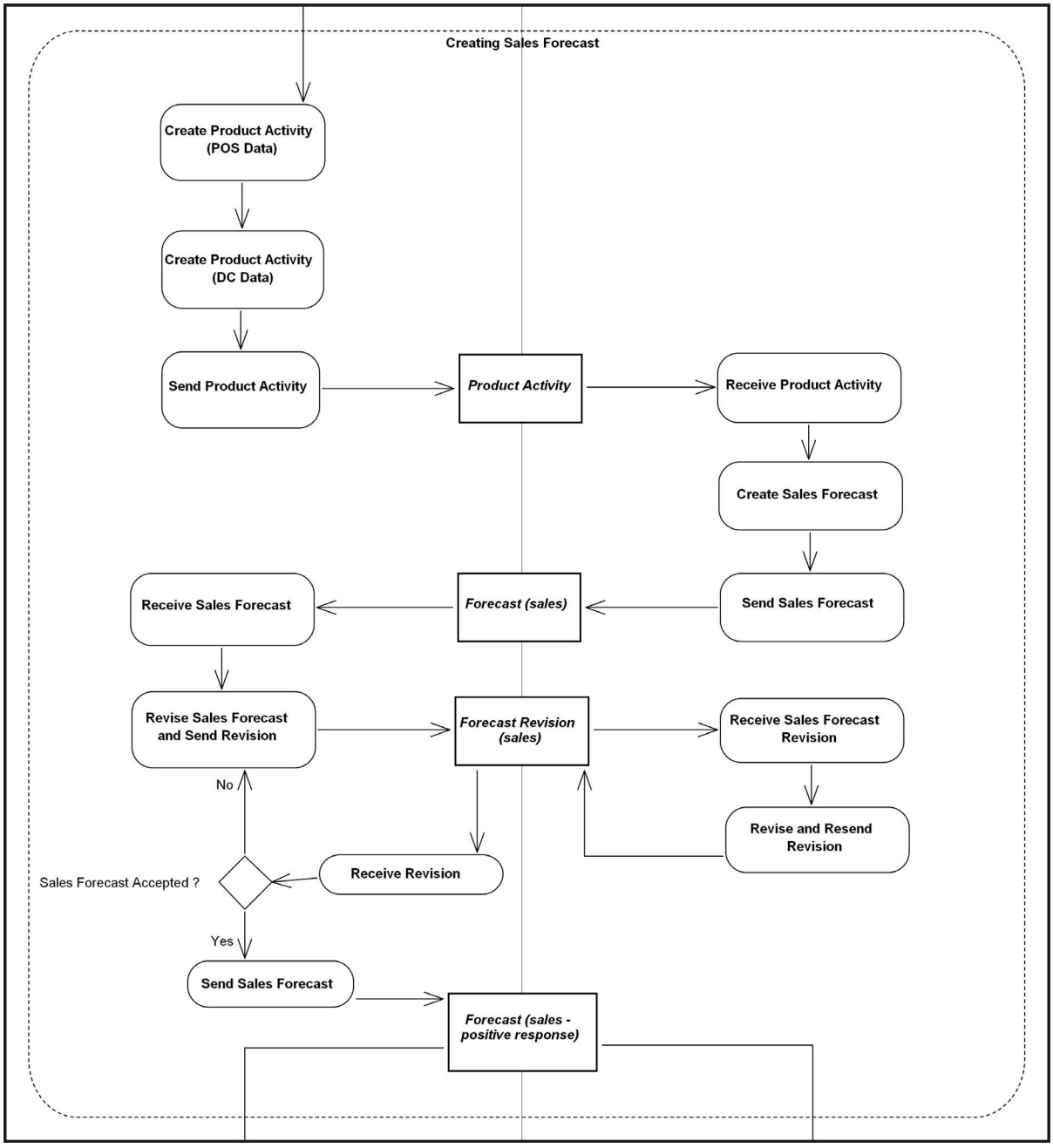
In CPFR Step 3, the Sales Forecast is generated. Following Option A, Conventional Order Management, from the CPFR implementation scenarios (see [**CPFRoverview**], Table 3), the responsible partner for the generation of Sales Forecast is the Seller. Having Event Calendar information and the Delivery Plan already in their system, there are two more kinds of information that the Seller needs for an effective Sales Forecast: POS Data and DC Data. As shown in Figure 32, “CPFR Steps 3, 4, and 5” and Figure 33, “Create Sales Forecast”, both of these pieces of information are sent within a Product Activity message. This time there is no revision of the messages because these messages contain statistical and historical information collected previously by the Buyer.

Figure 32. CPFR Steps 3, 4, and 5



Based on the event details (dates, products, tactics, etc.) and using the available data source(s), a volume estimate/forecast is created for each product/store combination included in the scope of the event by the Seller. During the calculation, sales forecasting algorithms make use of the coefficients for causal factors based on the event history. Once the Sales Forecast suggestion is generated and sent to the Buyer, the Buyer revises it and might recommend some changes on the Forecast. The Forecast Revision message exchange continues until the forecast is agreed by both sides.

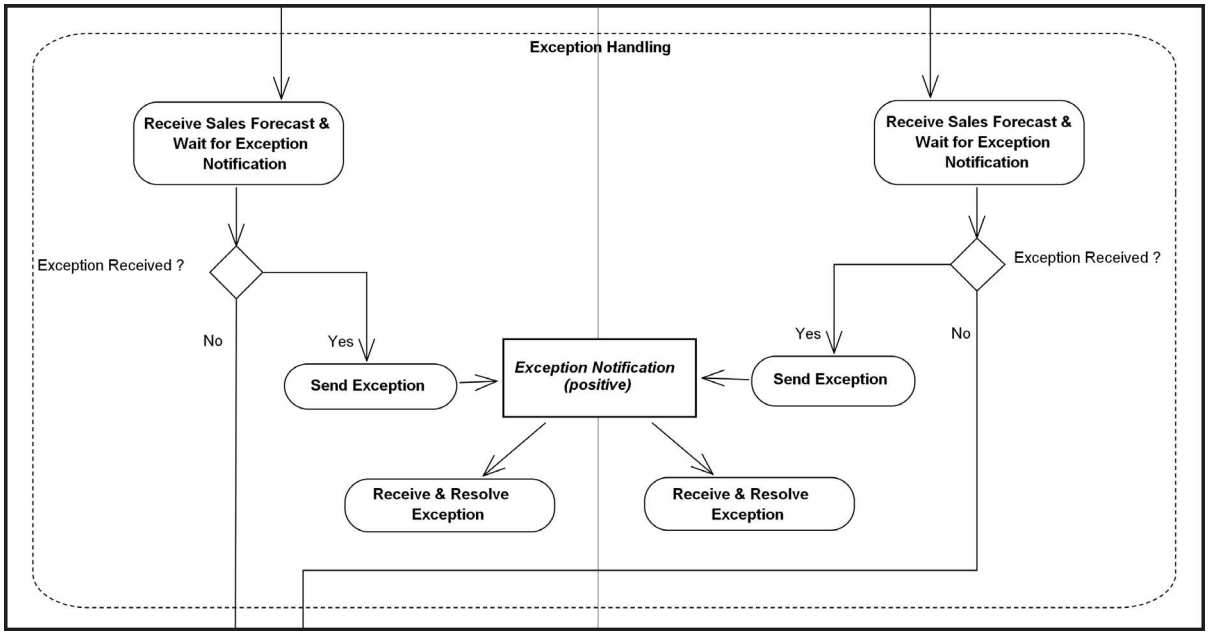
Figure 33. Create Sales Forecast



On average, six weeks elapse between Sales Forecast Generation and Order Generation. During this period, both sides observe changes to the conditions. If one of the partners detects an exception invalidating the exception criteria defined in CPFR Step 1, it sends an Exception Notification message to the other party. Exceptional circumstances that may be communicated between trading partners include deviations between planned impacts (either between buyer and seller, or between subsequent generations of planned impacts from the same trading partner), as well as deviations between planned and actual impacts. It should be noted that both sides might detect an exception, and therefore both sides should be capable of sending and receiving exceptions. Of course, for specific implementations if the collaborating parties want to change this behaviour, they can customize the process so that one partner will be responsible for the generation of the Exception Notifications.

CPFR Step 4 is solely composed of the exception generation and receiving activity. CPFR Step 5, on the other hand, is the resolution of the Exceptions.

Figure 34. Exception Handling

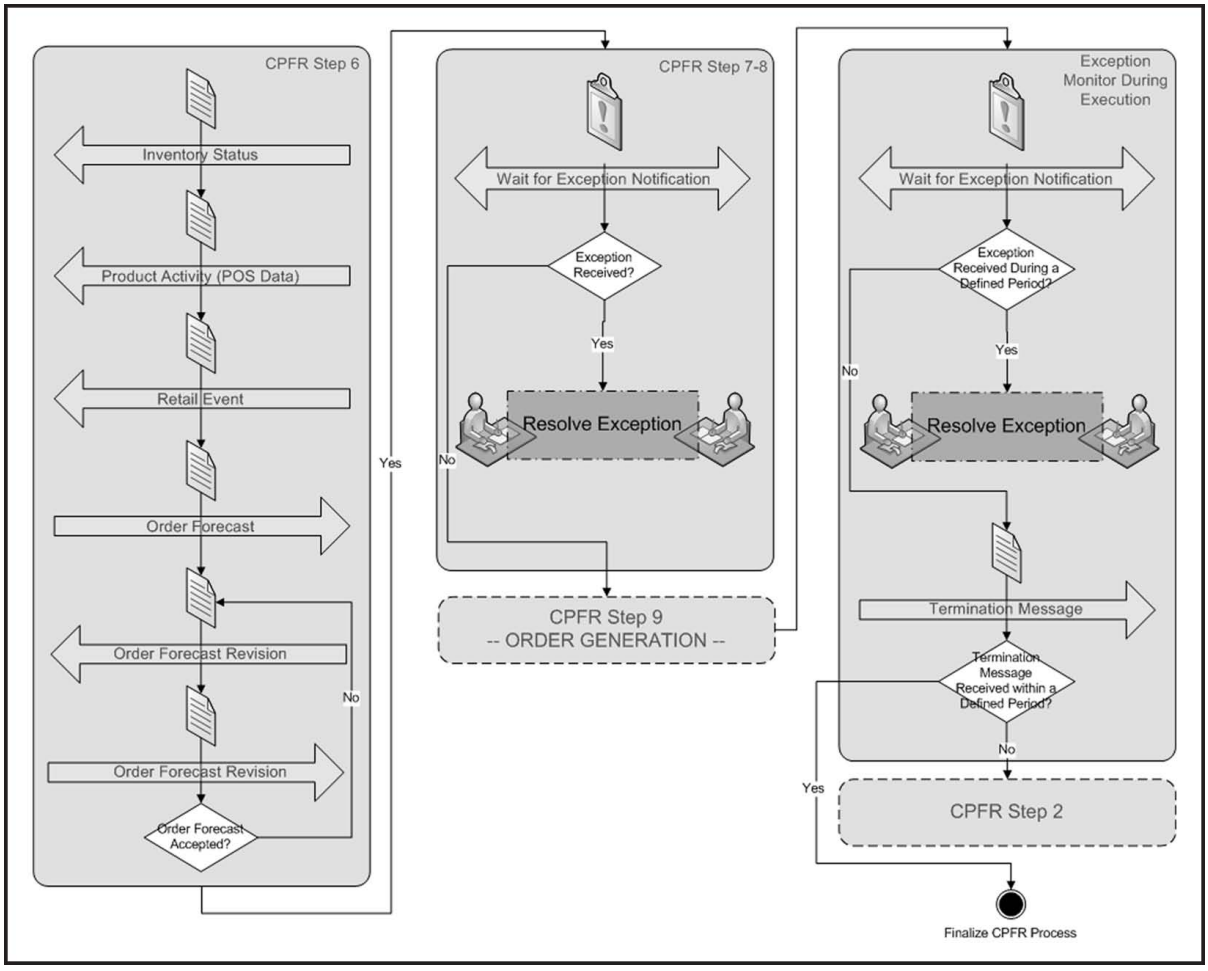


If there is no Exception Notification Message within the defined period, the process continues with Order Forecast Generation (CPFR Step 6).

2.13.3 Order Forecast Generation and Exception Handling

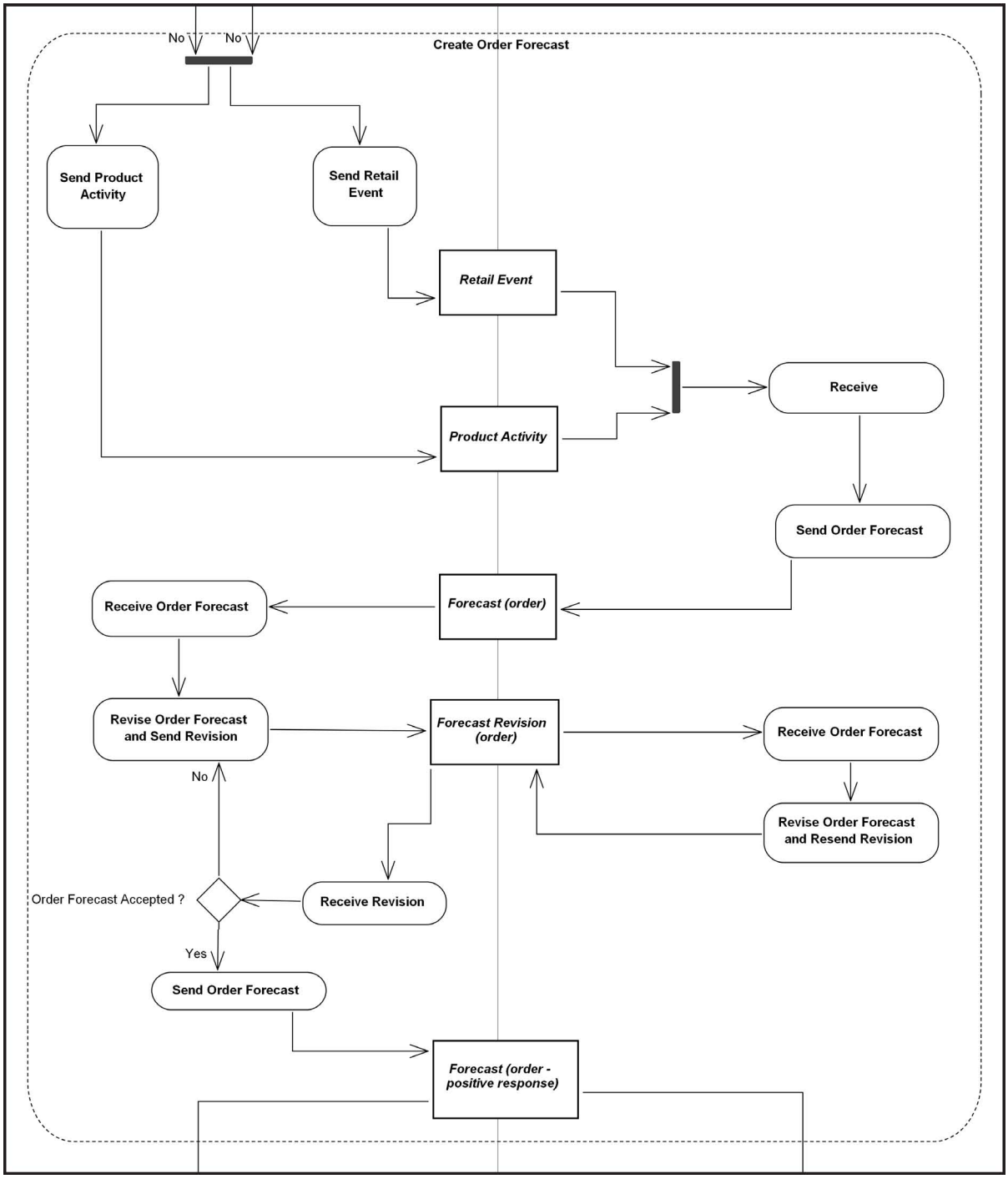
In the supply chain process, it is important for sales forecasts that are created to be converted into the shipment (order) forecasts that can then be used in the production planning processes at the manufacturing locations and be incorporated into the ordering processes at the retailer. As shown in Figure 35, “CPFR Steps 6, 7, 8 and 9”, the responsibility for creating Order Forecast belongs to the Seller per Option A of the CPFR implementation scenarios (see [**CPFRoverview**], Table 3). Sales forecasts can be transformed into order forecasts by incorporating inventory status information, possible retail event plans, and current point of sale data. Therefore, Buyer sends the updated versions of the Retail Event, Inventory Status, and POS Data to the Seller.

Figure 35. CPFR Steps 6, 7, 8 and 9



After the Seller creates the Order Forecast using the obtained data, it sends the forecast to the Buyer. The Buyer checks the order forecast and sends back a revision document which includes update requests if necessary. The exchange of Order Forecast Revisions continues until there are no further update requests and the Order Forecast is agreed by both sides. Document types used in this process are Retail Event, Product Activity, Forecast, and Forecast Revision.

Figure 36. Create Order Forecast



After the Order Forecast is frozen, the process continues with the exception detection activity (CPFR Step 7). The exception detection process that follows Order Forecast is similar to process described earlier for exception detection following Sales Forecast (see Section 2.13.2, “Sales Forecast Generation and Exception Handling”). The only difference between the Order Forecast and Sales Forecast exceptions is the content of the exceptions.

CPFR Step 8, Order Forecast Exception Resolution activity, is handled similarly to Sales Forecast Exception Resolution.

Figure 37. Identifying and Resolving Exceptions for Order Forecast

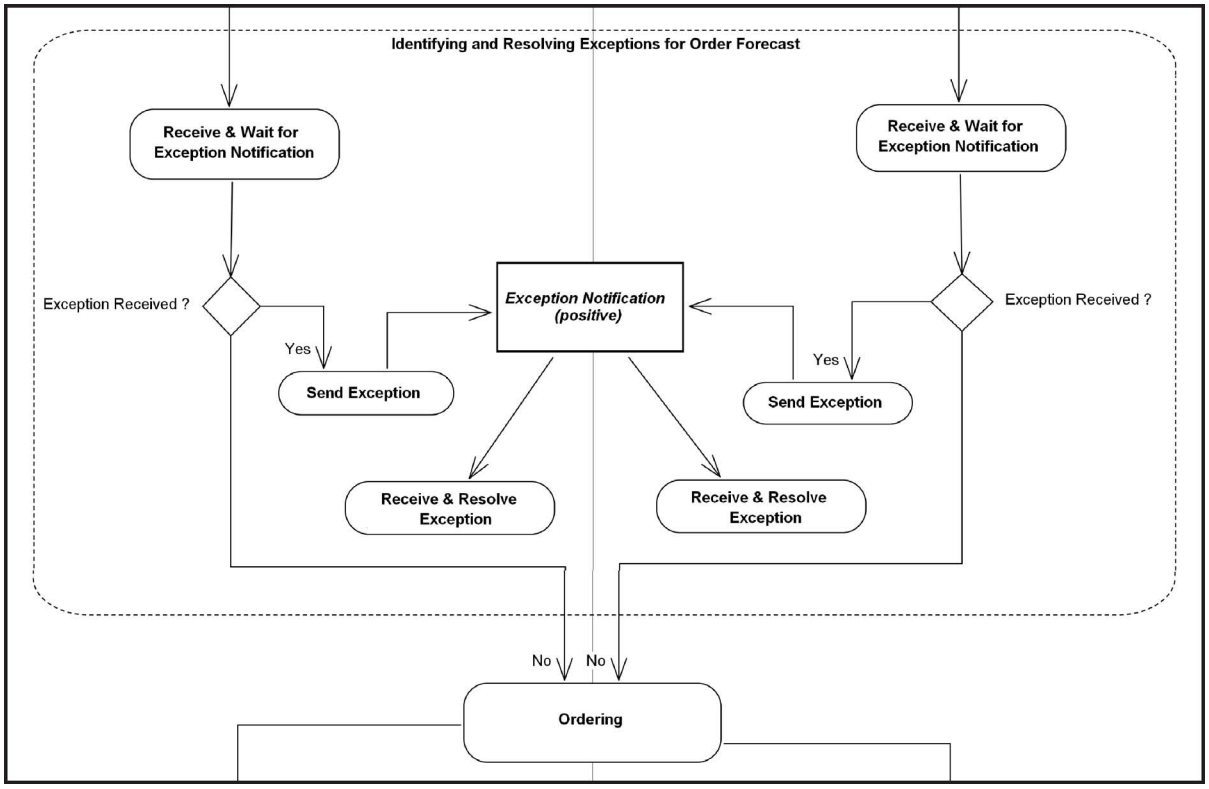
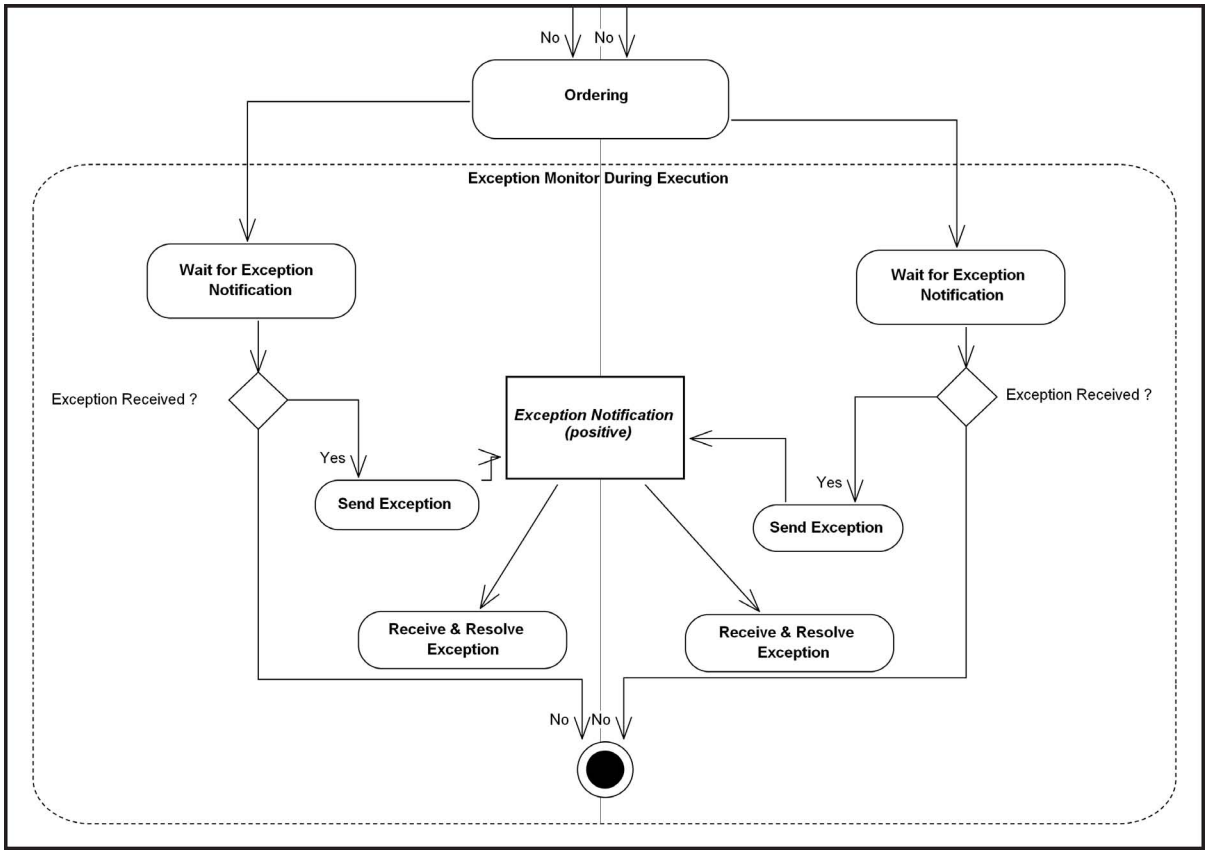


Figure 38. Exception Monitor During Execution



If there is no exception during a period of time, the process continues with the Order Generation Step.

From the technical point of view, the exception monitoring and its resolution are exactly same as in the case of Order Forecast Exception Handling and Sales Forecast Exception Handling. The difference is in the content of the exceptions. The actual events and orders are compared to the Forecasted Sales and Forecasted Orders. When there is a situation violating the normal exception criteria, one of the sides might generate an exception notification. Besides comparison of forecasts, other information gathered during the execution is observed (e.g., event dates, POS data, etc.). The resolution of the exceptions is the same as the process carried out for Sales Forecast Exception resolution.

2.14 Vendor Managed Inventory

Vendor Managed Inventory (VMI) is a family of business processes in which the Retailer Customer Party for an item provides certain information to the Seller Supplier Party, and the Seller Supplier Party takes full responsibility for maintaining an agreed-upon inventory of the item, usually at the Retailer Customer Party's point of sale. A third party logistics provider can also be involved to make sure that the Retailer Customer Party has the required level of inventory by adjusting the demand and supply gaps.

UBL supports three common models of VMI:

- Basic VMI
- Cyclic Replenishment Program (CRP)
- Replenishment on Customer Demand

These processes are described in more detail below. It should be noted that the particular semantics used here come from a large-scale UBL application developed for the Italian textile and clothing industry by ENEA, the Italian National Agency for New Technologies, Energy, and Sustainable Economic Development (see [**eBiz-TCF**]). These models are applicable to the implementation of vendor-managed relationships in a broad range of retail sectors, but for the sake of simplicity, and in keeping with the model application, the two principal parties in the VMI relationship (the Seller Supplier Party and the Retailer Customer Party) are referred to as "producer" and "retailer" in the descriptions that follow; more generically, they are vendor and customer.

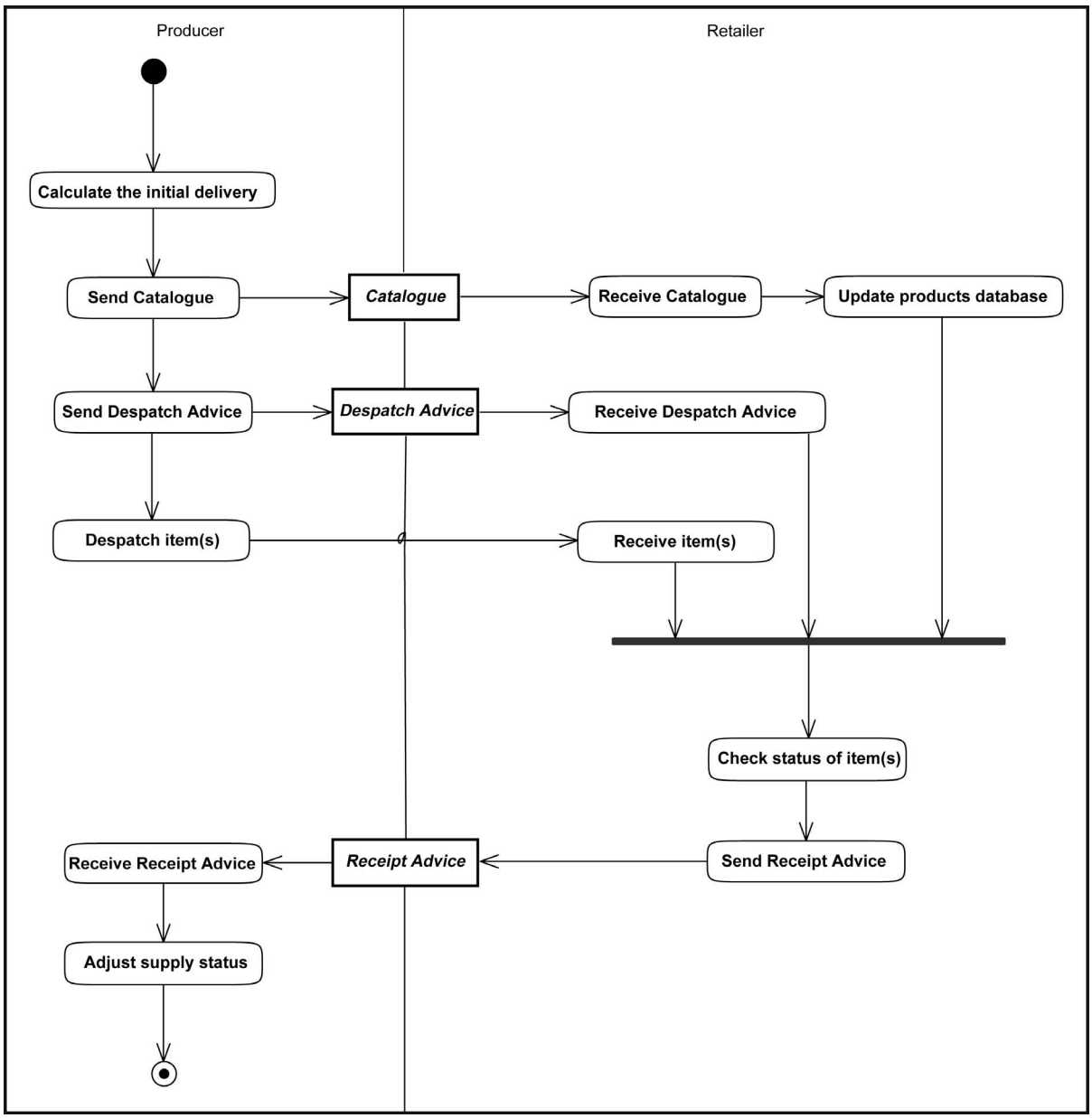
2.14.1 Basic Vendor Managed Inventory

In the classic VMI scenario, a shop-within-a-shop area or an entire store is managed completely by the producer. The logistic concept of VMI can be combined with consignment/concession as well as with charge-on-delivery as the financial model. Mostly it is combined with consignment.

2.14.1.1 Initial Stocking of the Area by Producer

At the beginning of the cooperation, the area is stocked by the producer. The retailer receives item and delivery information and reports back the goods actually received. UBL document types used here are Catalogue, Despatch Advice, and Receipt Advice.

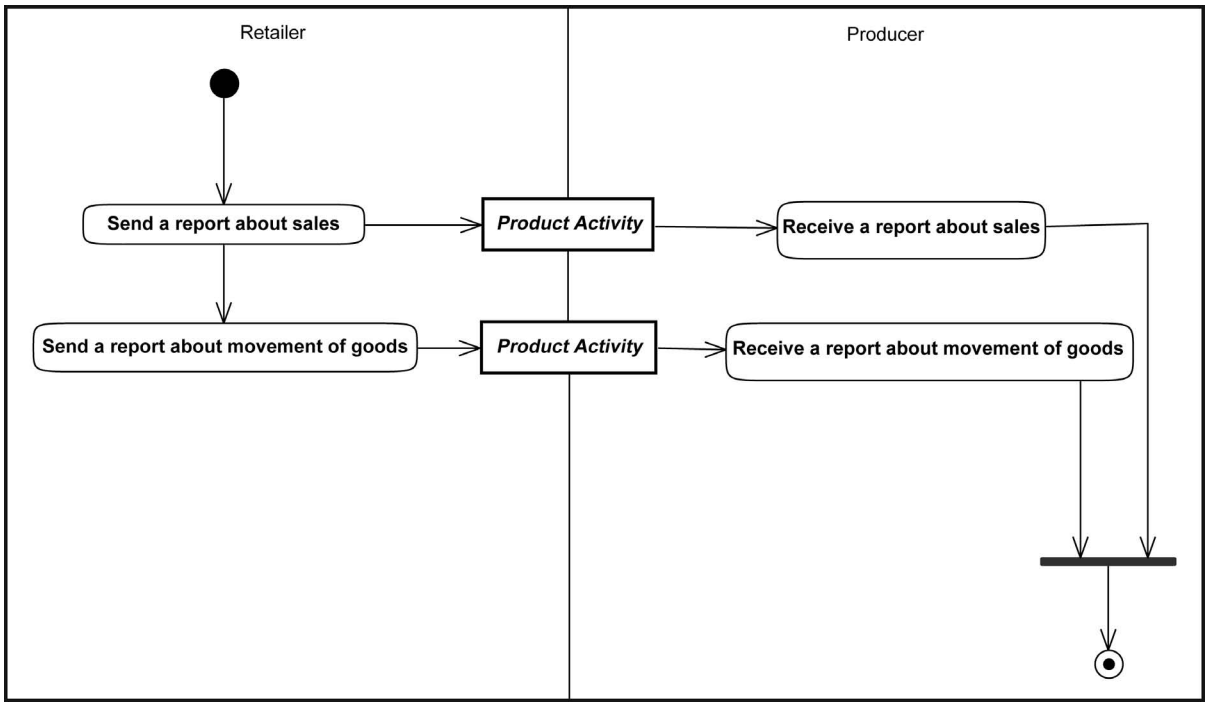
Figure 39. Initial Stocking of the Area by Producer



2.14.1.2 Report of Sales and Inventory Movement

The sales and inventory movement information is transferred from the retailer to the producer using Product Activity.

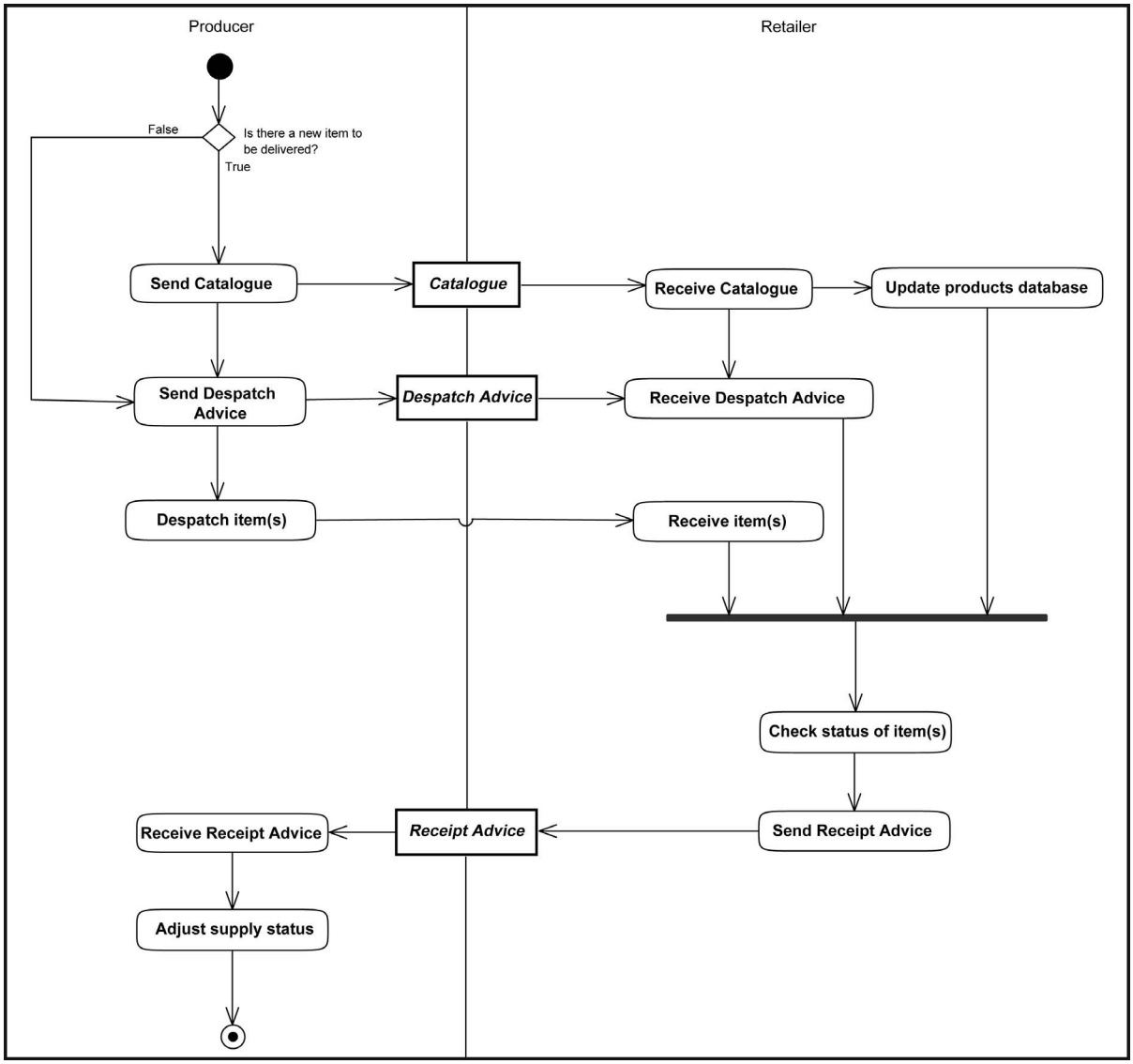
Figure 40. Report of Sales and Inventory Movement



2.14.1.3 Permanent Replenishment

Based on sales and inventory movement, the producer periodically makes a new delivery of goods accompanied by a Despatch Advice. If the delivery contains an item not previously stocked, an updated Catalogue is also sent so that the retailer can add the item to its product database. Upon delivery of the goods, the retailer reports back the items received using a Receipt Advice.

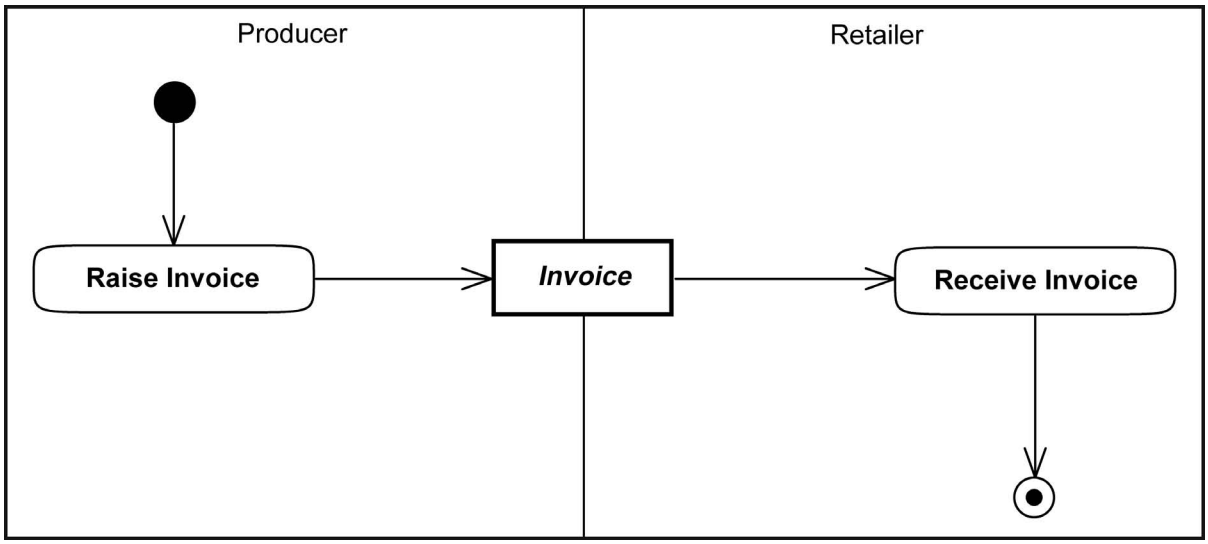
Figure 41. Permanent Replenishment



2.14.1.4 Invoicing for Vendor Managed Inventory

A UBL Invoice is sent either on a delivery or a sales basis. In a charge-on-delivery model, the data for the invoice is prepared from the delivery, and in a consignment/concession model from the sales reports.

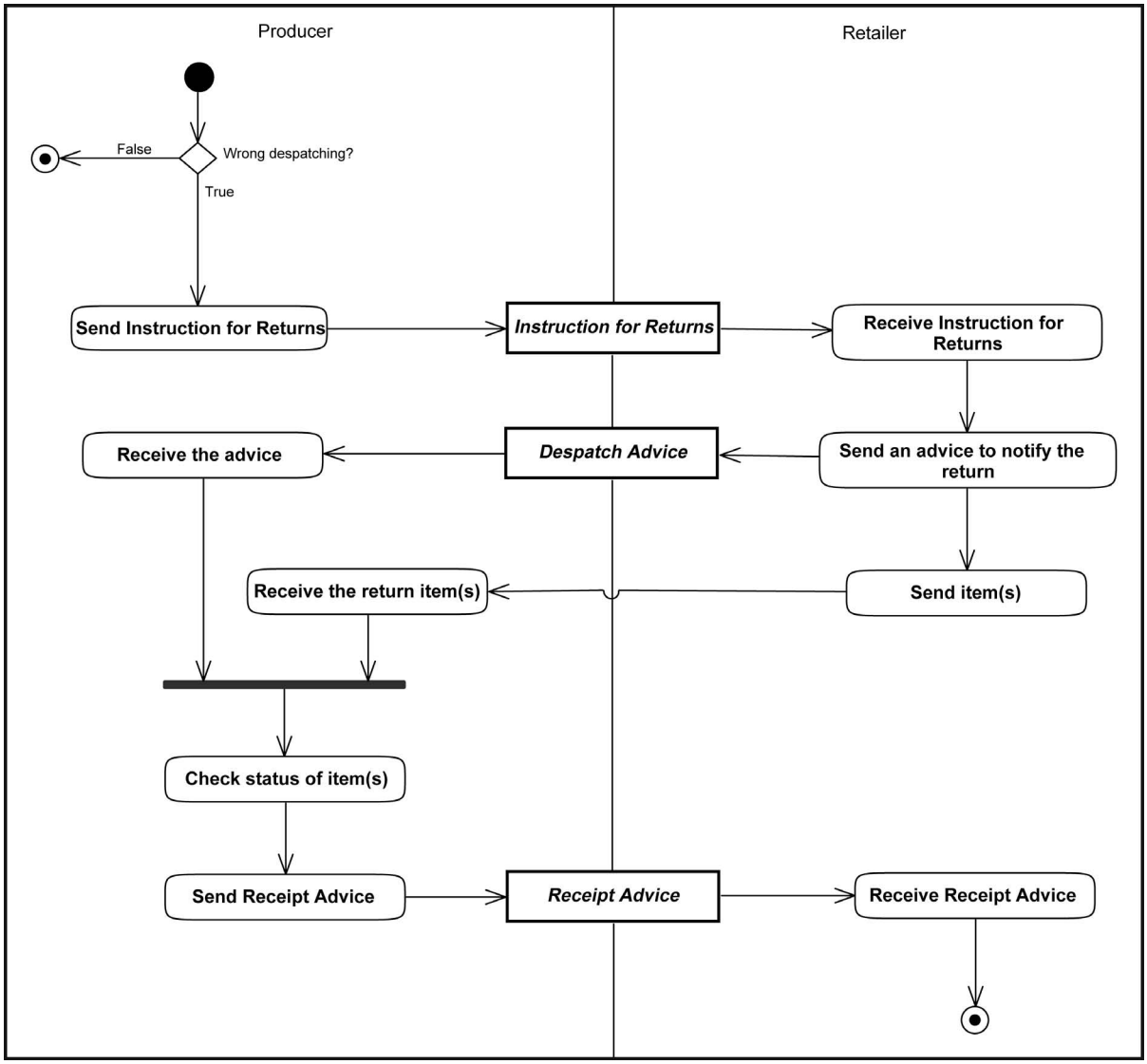
Figure 42. Invoicing for Vendor Managed Inventory



2.14.1.5 Returns Initiated by the Producer

If sales do not meet expectations, items are reallocated by the producer. Because the producer cannot request a retailer to send the products to a competitor, the producer requests a return and handles the goods afterwards by itself. Document types used here are Instruction for Returns, Despatch Advice, and Receipt Advice.

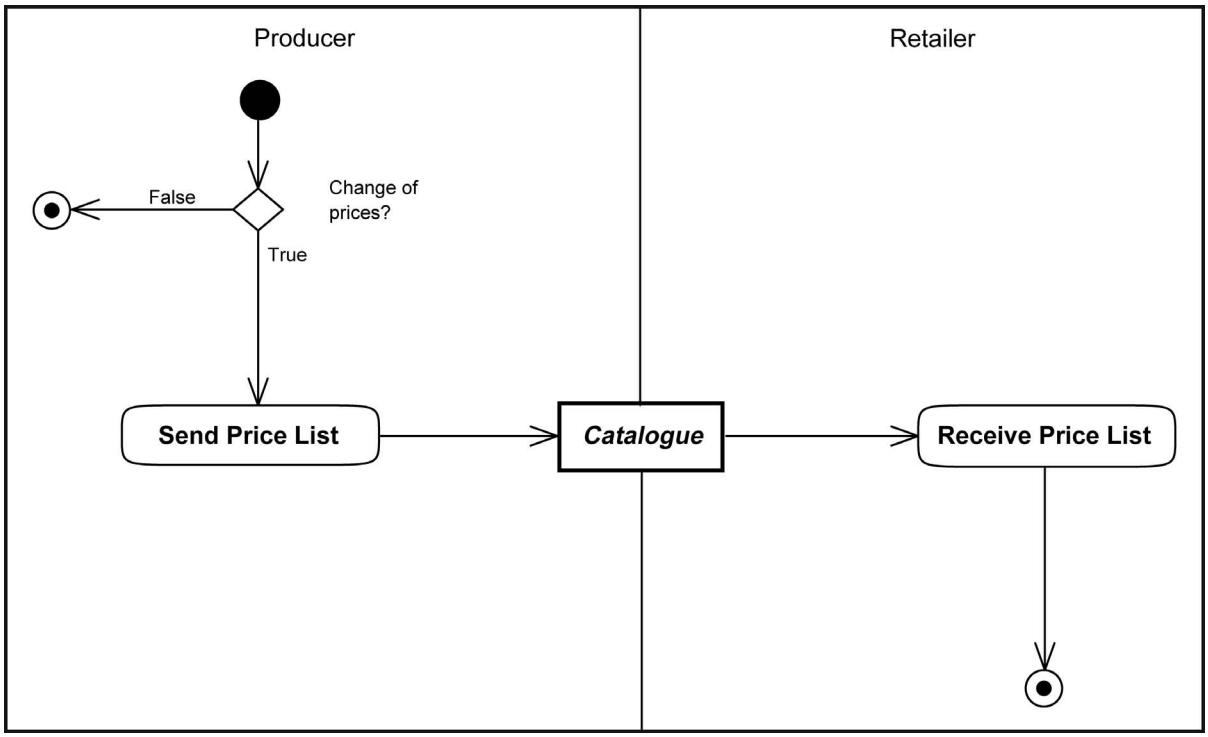
Figure 43. Returns Initiated by the Producer



2.14.1.6 Price Adjustments

In the event of a price change, an updated price list (in the form of a new Catalogue containing the change) is sent from producer to retailer.

Figure 44. Price Adjustments



2.14.2 Cyclic Replenishment Program (CRP)

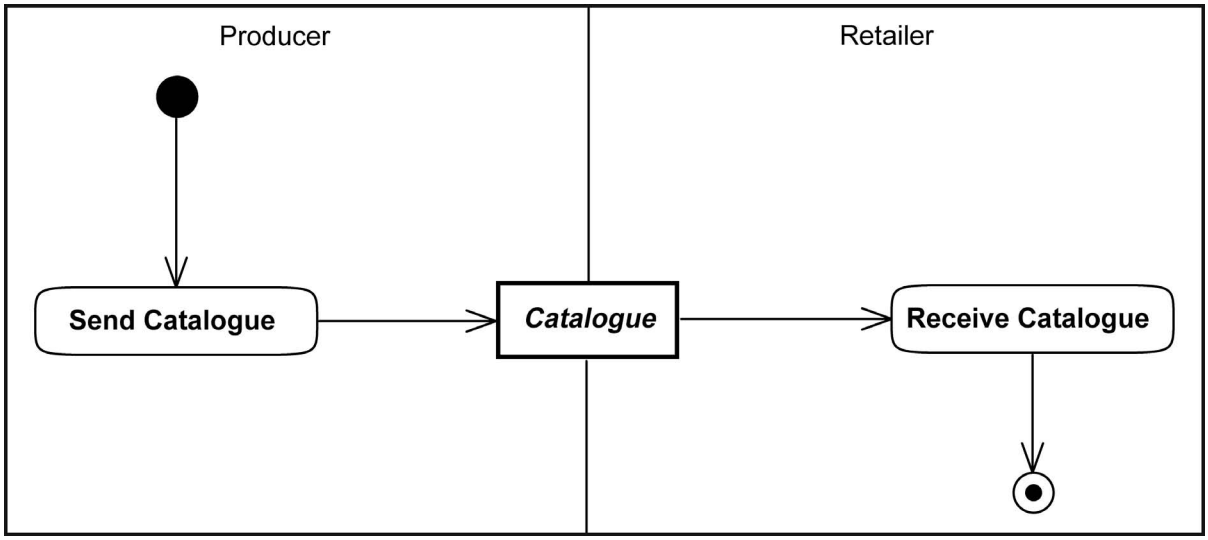
A variant of VMI is the Cyclic Replenishment Program (CRP). In this process, the producer establishes a catalogue of NOS (Never Out of Stock) or seasonal NOS items, and the retailer chooses items for cyclic (weekly) replenishment. The logistic scenario can be combined with the charge-on-delivery as well as with a consignment/concession model. At the end of every sales period, a report of sales and inventory movement at all retail locations is sent to the producer.

CRP differs from the third VMI variant, Replenishment on Customer Demand (below), in that the producer cannot change the terms of the order.

2.14.2.1 Transfer of Base Item Catalogue

The producer publishes the Catalogue of its NOS and seasonal NOS items to the retailer.

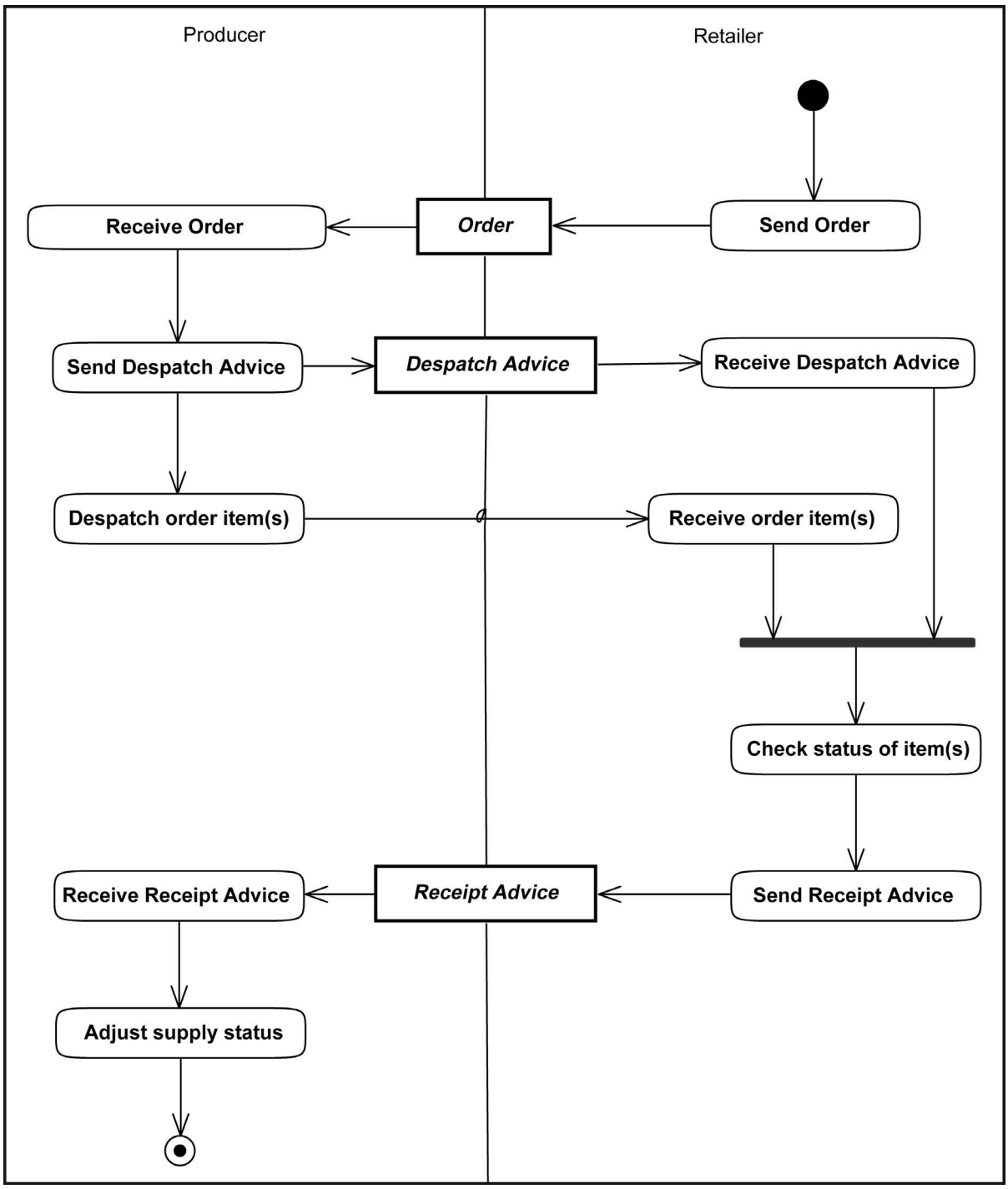
Figure 45. Transfer of Base Item Catalogue



2.14.2.2 Initial Stocking of the Area by Retailer

At the beginning of the cooperative relationship—or the beginning of a season, if seasonal NOS products are the focus—the retailer orders its base stock, and the products are delivered. Order, Despatch Advice, and Receipt Advice are used in this process.

Figure 46. Initial Stocking of the Area by Retailer



2.14.2.3 Periodic (Weekly) Replenishment

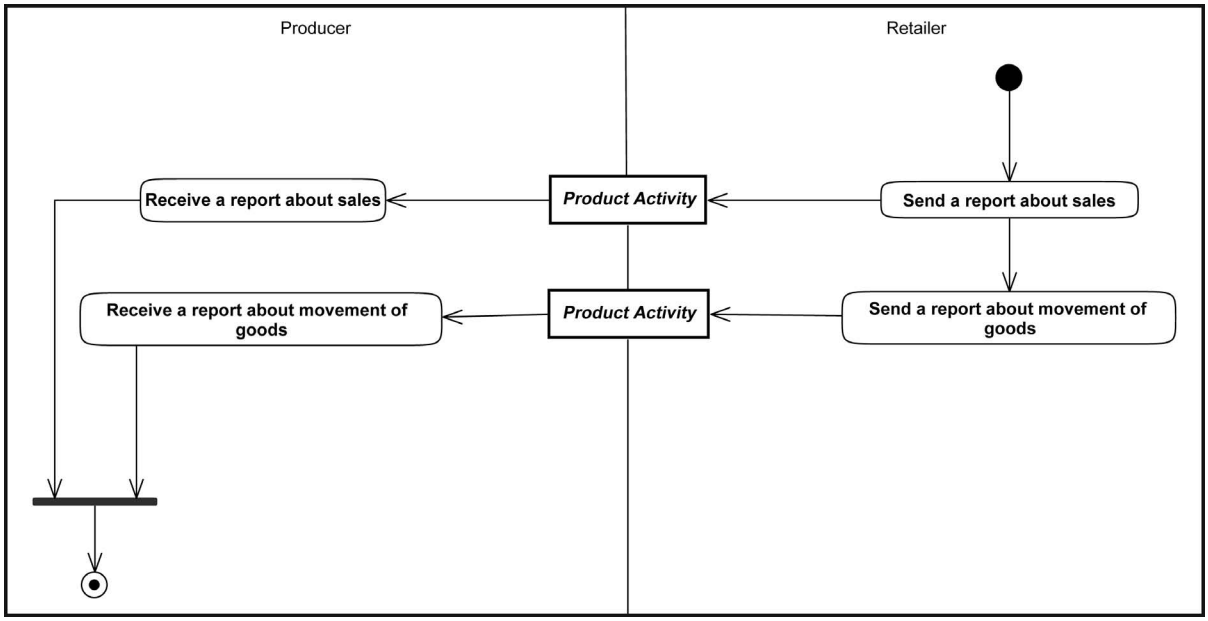
Each period (every week), the retailer's system calculates the quantities needed for replenishment of the product area. From the result, an order is sent, and the producer responds with a direct delivery within 48 hours.

The replenishment process uses the same documents in the same order as the Initial Stocking process, so the duplicate diagram is omitted here; see Figure 46, “Initial Stocking of the Area by Retailer”. It must be remembered, however, that the two processes are taking place at different points in time, so their pre and post conditions will be different.

2.14.2.4 Report of Sales and Inventory Movements

At the end of each sales day, a report of all sales and inventory movement at all retail locations is sent from the retailer to the producer using Product Activity.

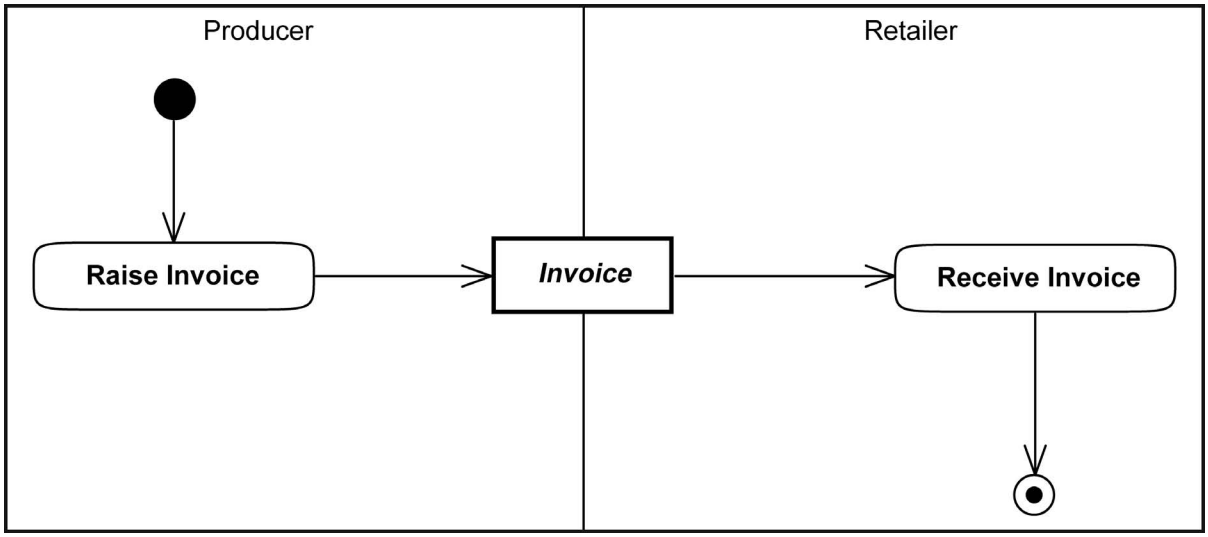
Figure 47. Report of Sales and Inventory Movements



2.14.2.5 Cyclic Replenishment Program Invoicing

A UBL Invoice is sent either on a delivery or a sales basis.

Figure 48. Invoicing for Cyclic Replenishment Program

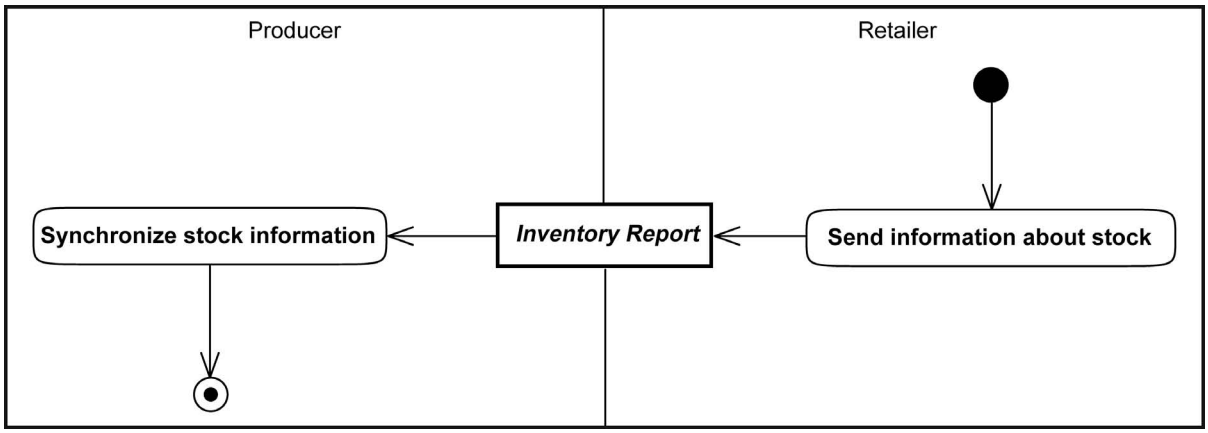


2.14.2.6 Synchronizing of Stock Information

Information about the actual stock is synchronised periodically (for example, every one to three months) using Inventory Report. This is combined at least once a year with a physical inventory.

The retailer sends an inventory report containing the information about the quantities currently in stock.

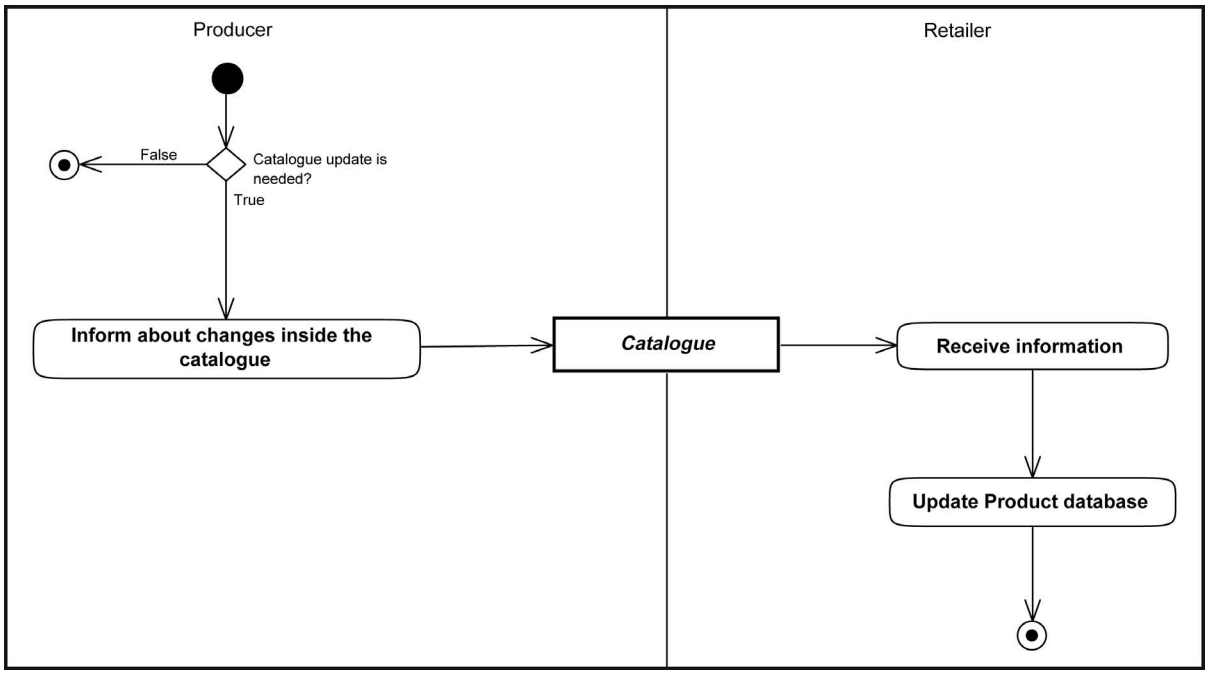
Figure 49. Synchronizing Stock Information



2.14.2.7 Changes to the Item Catalogue

In the event of a change, either inside an item belonging to the CRP Catalogue or the relationship of an item to the CRP Catalogue, information about the change is sent to the retailer by sending an updated Catalogue document. Item change is indicated by an optional ActionCode field in each changed CatalogueLine.

Figure 50. Changes to the Item Catalogue



2.14.3 Replenishment On Customer Demand

Another variant of VMI is Replenishment On Customer Demand. In this process, the producer selects a subset of its products for a specific retailer and sends out the related article catalogue. Then the pro-

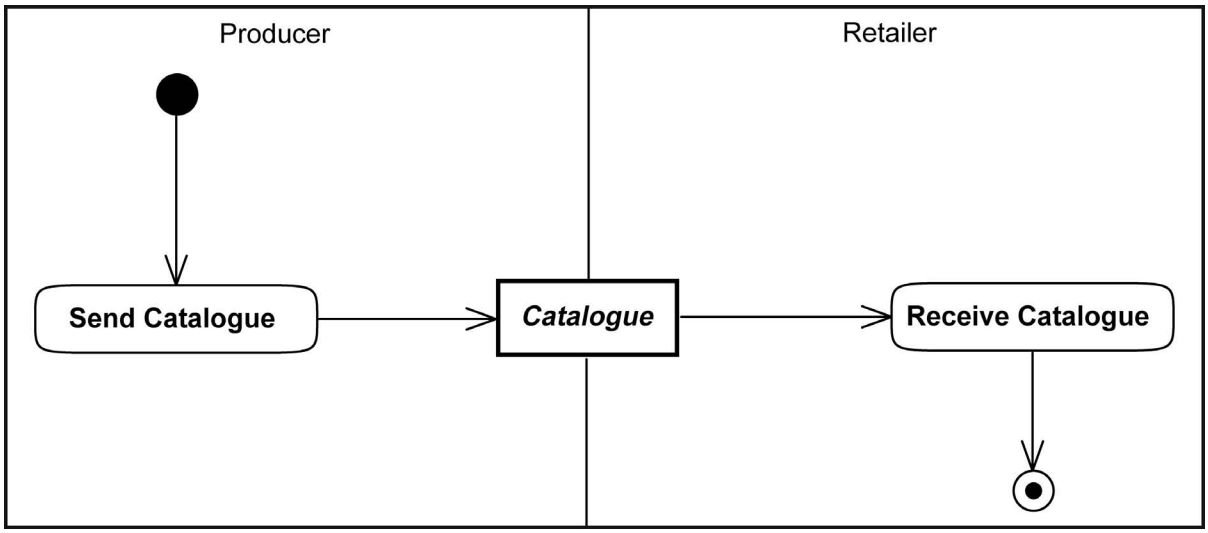
ducer periodically sends information about the availability of items so that the retailer can form the best ordering plan. The replenishment periodically happens on retailer (customer) demand, and unlike the case with CRP (above), the producer is allowed to propose changes to the orders. Also, because of the requirement to update item availability information, an additional document type (Stock Availability Report) is added to the process.

The processes of sales and inventory reporting, invoicing, stock synchronization, and changing the catalogue are identical to the same processes in CRP. As with CRP, a report of sales and inventory movement at all retail locations is sent to the producer at the end of every sales period. Invoicing and logistics are normally charge-on-delivery but can also be based on a consignment/concession model.

2.14.3.1 Transfer of Base Article Catalogue

The producer publishes a Catalogue of its products to the retailer. The catalogue can include basic articles, never-out-of-stock (NOS) articles, seasonal articles, short-season-collection articles, or seasonal NOS articles.

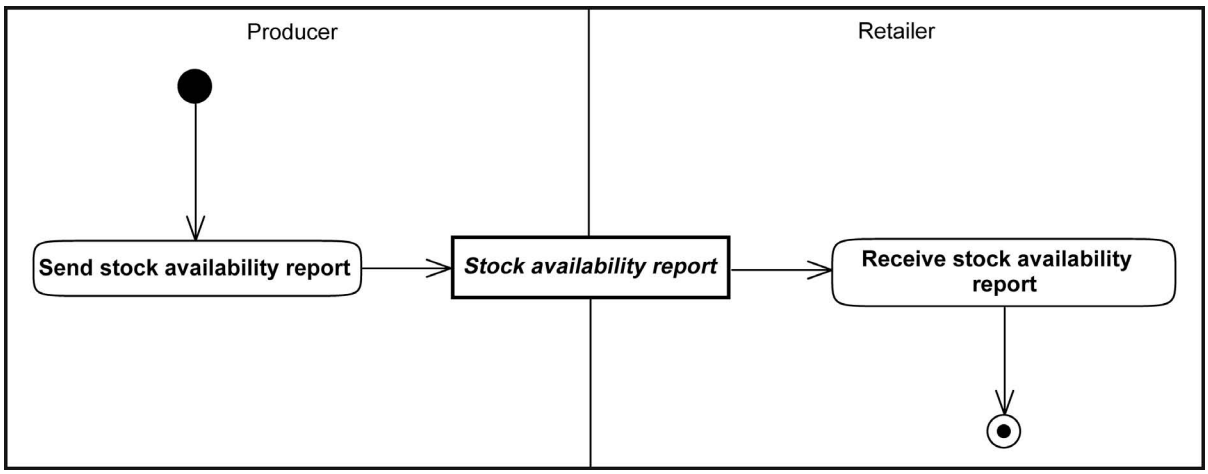
Figure 51. Transfer of Base Article Catalogue



2.14.3.2 Periodic Transfer of Article Availability Information

The producer sends out information about availability of goods (quantities on hand, quantities incoming, articles out of stock) using a Stock Availability Report.

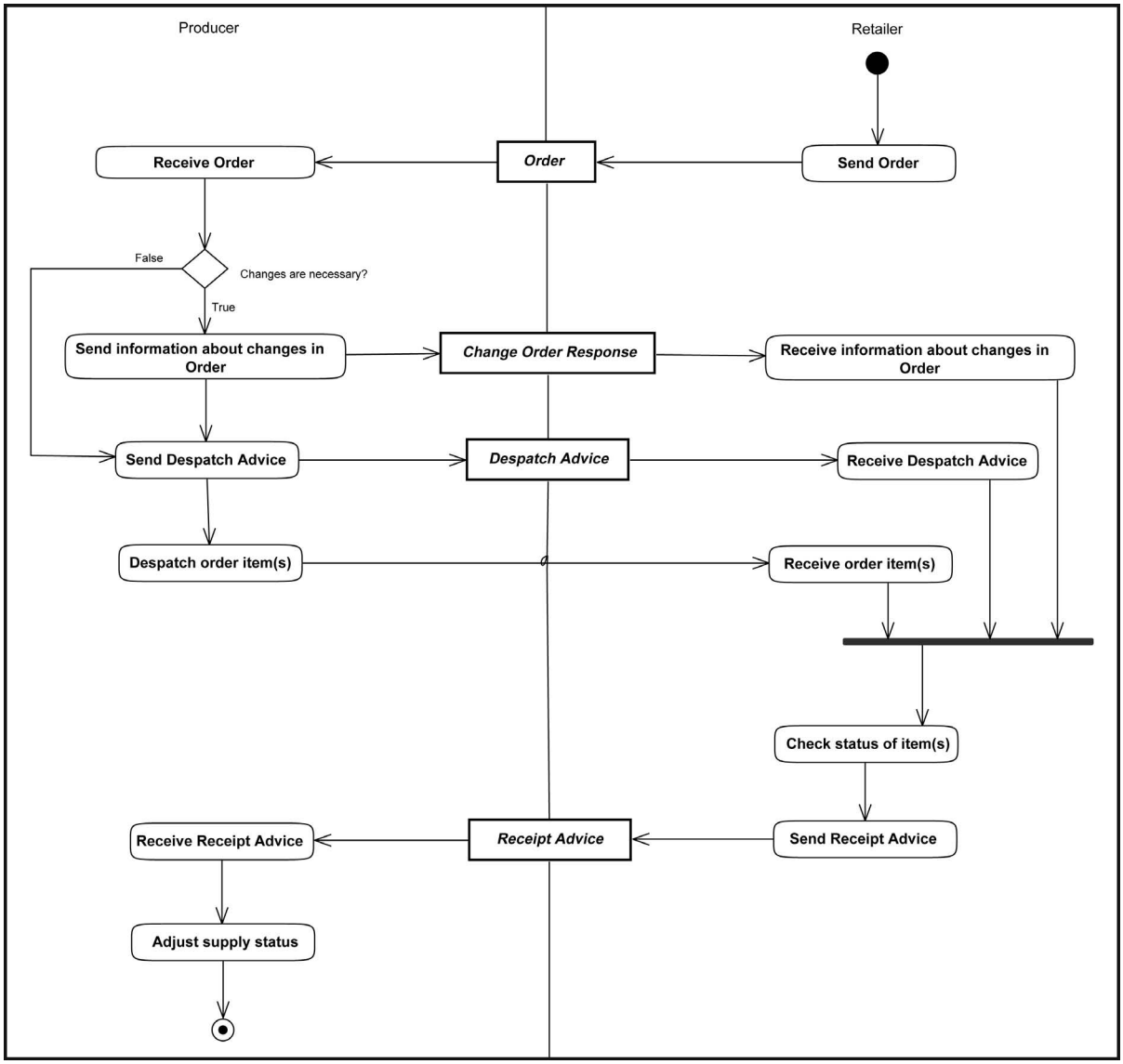
Figure 52. Periodic Transfer of Article Availability Information



2.14.3.3 Initial Stocking of the Area by Producer and Retailer

At the beginning of the business cooperation—or perhaps at the beginning of a season, if seasonal NOS (never out of stock) products are the focus—the retailer orders its base stock and the products are delivered. Note that the producer is allowed to propose changes to the order (compare this figure with Figure 46, "Initial Stocking of the Area by Retailer"). Document types used in this process include Order, Order Change, Despatch Advice, and Receipt Advice.

Figure 53. Initial Stocking of the Area by Producer and Retailer



2.14.3.4 Periodic Replenishment

Periodically, the retailer's system calculates the quantities needed for replenishment of the area. From the result, an order is sent, and the producer responds with a direct delivery within 48 hours.

The replenishment process uses the same documents in the same order as the Initial Stocking process, so the duplicate diagram is omitted here; see Section 2.14.3.3, "Initial Stocking of the Area by Producer and Retailer". It must be remembered, however, that the two processes are taking place at different points in time, so their pre and post conditions will be different.

2.14.3.5 Report of Sales and Inventory Movement

Sales and inventory movement information is transferred daily from the retailer to the producer.

The process for sales and inventory reporting is the same as in CRP (see Figure 47, “Report of Sales and Inventory Movements”).

2.14.3.6 Invoicing for Replenishment On Customer Demand

An invoice is sent either on a delivery or a sales basis.

The invoice process for Replenishment On Customer Demand is the same as for CRP (see Figure 48, “Invoicing for Cyclic Replenishment Program”).

2.14.3.7 Synchronizing Stock Information

Information about the actual stock is synchronised periodically (for example, every one to three months). Synchronization occurs at least once a year together with a physical inventory.

The stock synchronization process for Replenishment On Customer Demand is the same as in CRP (see Figure 49, “Synchronizing Stock Information”).

2.14.3.8 Changes to the Article Catalogue

In the event of a change, either inside an item belonging to the Catalogue or the relationship of an item to the Catalogue, information about the change is sent to the retailer by sending an updated Catalogue document. Item change is indicated by an optional ActionCode field in each changed CatalogueLine.

The process for changing the catalogue in Replenishment On Customer Demand is the same as in CRP (see Figure 50, “Changes to the Item Catalogue”).

2.15 International Freight Management

Freight management for domestic trade is typically accomplished using Despatch Advice and Receipt Advice (see Section 2.7, “Fulfilment”). The additional processes shown in Figure 54, “Initiate Freight Management Process” are engineered to support the ordering and management of logistical services for international trade.

With receipt of an order and acknowledgement by the Supplier Party that the goods are available and ready to be shipped, the Consignor or Consignee initiates the transportation arrangements. This includes booking the consignment with a Transport Service Provider such as the Freight Forwarder or Carrier and advising the Delivery Party of the arrangements as needed.

Document types in these processes are Forwarding Instructions, Packing List, Bill of Lading, and Waybill. (Regarding the Transportation Status document type, see Section 2.16, “Freight Status Reporting”).

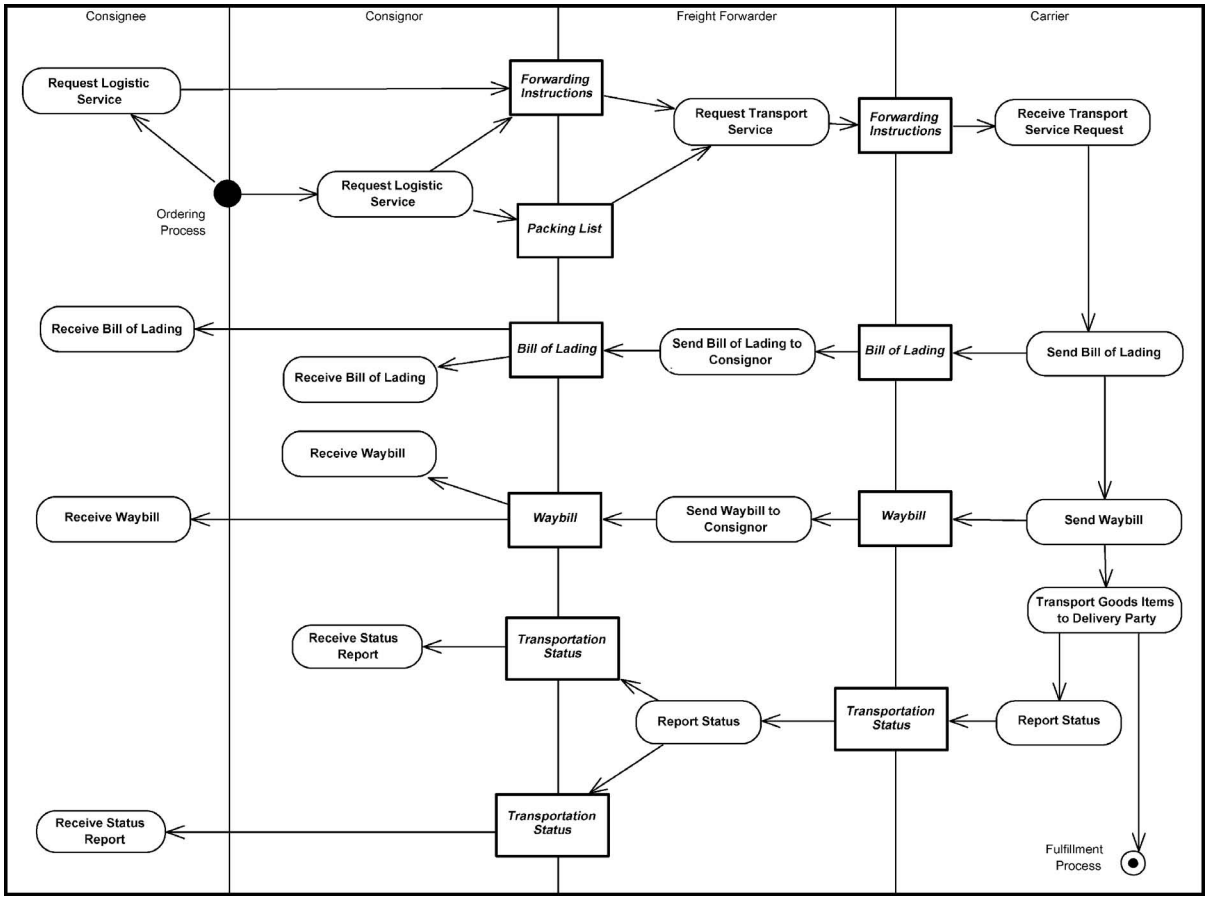
It should be noted that these processes involve the Consignee and Consignor and do not cover all the logistical processes required to physically move the goods or regulatory notifications such as Customs declarations.

Note

For a discussion of the difference between *consignment* (consignor to consignee) and *shipment* (shipper to recipient) see Section D.2, “Shipment vs. Consignment”.)

For a discussion of the difference between *transport* and *transportation* see Section D.3, “Transport vs. Transportation”.

Figure 54. Initiate Freight Management Process



2.15.1 Forwarding Instructions

Forwarding Instructions are normally used by any party who gives instructions for the transportation services required for a consignment of goods (the Transport Service Buyer) to any party who is contracted to provide the transportation services (called the Transport Service Provider). Forwarding Instructions may also be used by any party who requests a booking of shipment space to be made for the transportation services required for a consignment of goods to any party who will provide the underlying transportation services. The parties who issue this document are commonly referred to as the shipper, consignee, or consignor, while the parties who receive this document are forwarders, carriers, shipping agents, etc.

Forwarding Instructions may also be issued by a freight forwarder or shipping agent in their capacity as a Transport Service Buyer. This document may be used to arrange for the transportation:

- Of different types of goods or cargoes
- Whether containerized or non-containerized
- Through different modes of transport, and
- From any origin to any destination.

2.15.2 Packing List

A Packing List is normally issued by the Consignor. It states the distribution of goods in individual packages.

2.15.3 Bill of Lading

A Bill of Lading is a transport document that is the evidence of a contractual agreement between the parties for the transportation service. The document evidences a contract of carriage by sea and the acceptance of responsibility for the goods by the carrier, by which the carrier undertakes to deliver the goods against surrender of the document. The Bill of Lading (B/L) may serve as a document of title. A provision in the document that the goods are to be delivered to the order of a named person, or to order, or to bearer, constitutes such an undertaking.

A Bill of Lading is normally issued by the party who provides the physical transportation services (e.g., the maritime carrier) to the party who gives instructions for the transportation services (shipper, consignor, etc.) as a receipt for the cargo and sometimes of instructions, stating the details of the transportation, charges, and terms and conditions under which the transportation service is provided.

A Bill of Lading may also be issued by the party who acts as an agent for the carrier or other agents to the party who gives instructions for the transportation services (shipper, consignor, etc.) stating the details of the transportation, charges, and terms and conditions under which the transportation service is provided, but who does not provide the physical transportation service. In such case a Bill of Lading is signed “as agent”.

Much of the information contained in the Bill of Lading corresponds to the information on the Forwarding Instructions. It is used for ocean or inland waterways modes of transport.

A freight forwarder, who can be either a Transport Service Provider or a Transport Service User according to different circumstances and depending on the contractual interlocutor, can assume responsibility for the shipment with regards to the shipper and issue Bills of Lading as a common carrier, a contractual carrier or as a Non Vessel Operating Common Carrier (NVOCC). In such case, when the transportation is multimodal, it can provide a multimodal Bill of Lading.

2.15.4 Waybill

A Waybill is a transport document issued by the party who undertakes to provide transportation services, or undertakes to arrange for their provision to the party who gives instructions for the transportation services (shipper, consignor, etc.). It states the instructions for the beneficiary and may contain the details of the transportation, charges, and terms and conditions under which the transportation service is provided.

Unlike a Bill of Lading, a Waybill is not negotiable and cannot be assigned to a third party (endorsement). It may be issued as a cargo receipt and is not required to be surrendered at the destination in order to pick up the cargo. This may simplify the documentation procedures between a Transport Service Buyer and a Transport Service Provider, but using this document in combination with international payments (e.g. documentary credits) is not advisable.

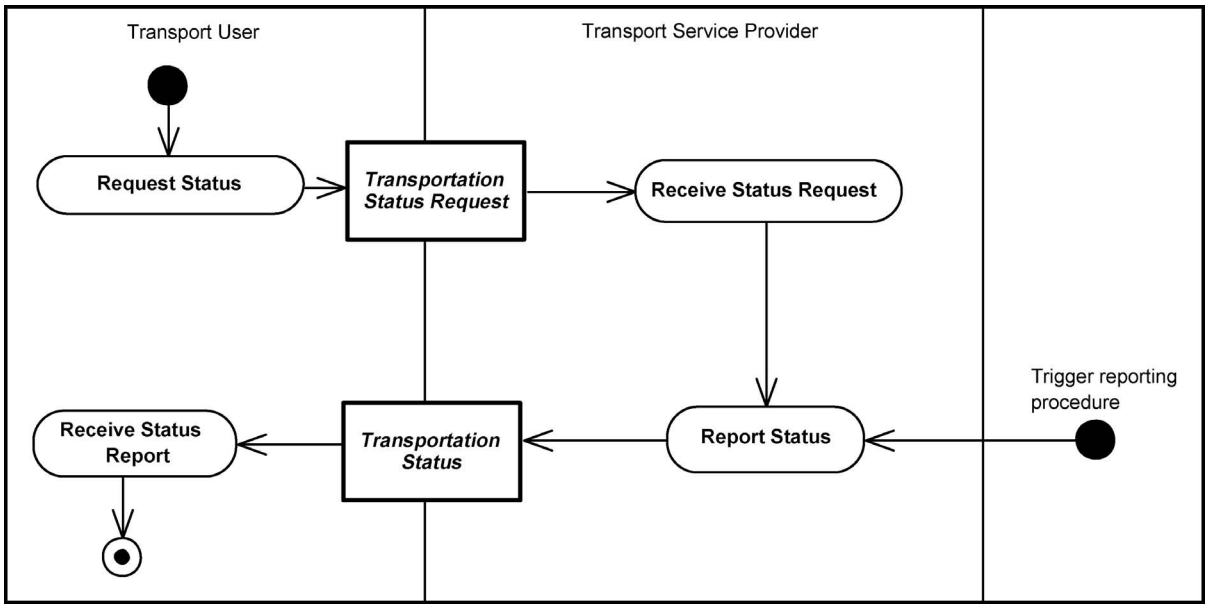
A freight forwarder may decide to issue a waybill to communicate consignment, transport, and conveyance information to third parties, be they shippers, subcontractors, transport operators, or authorities.

2.16 Freight Status Reporting

Freight Status Reporting is the process by which a Transport Service Provider (such as a Carrier or Freight Forwarder) communicates the status of shipments currently under their management to the Transport Users (such as a Freight Forwarder, Consignee, or Consignor).

A Transportation Status document is provided either through a Transportation Status Request document or through an agreed status reporting procedure.

Figure 55. Freight Status Reporting Process

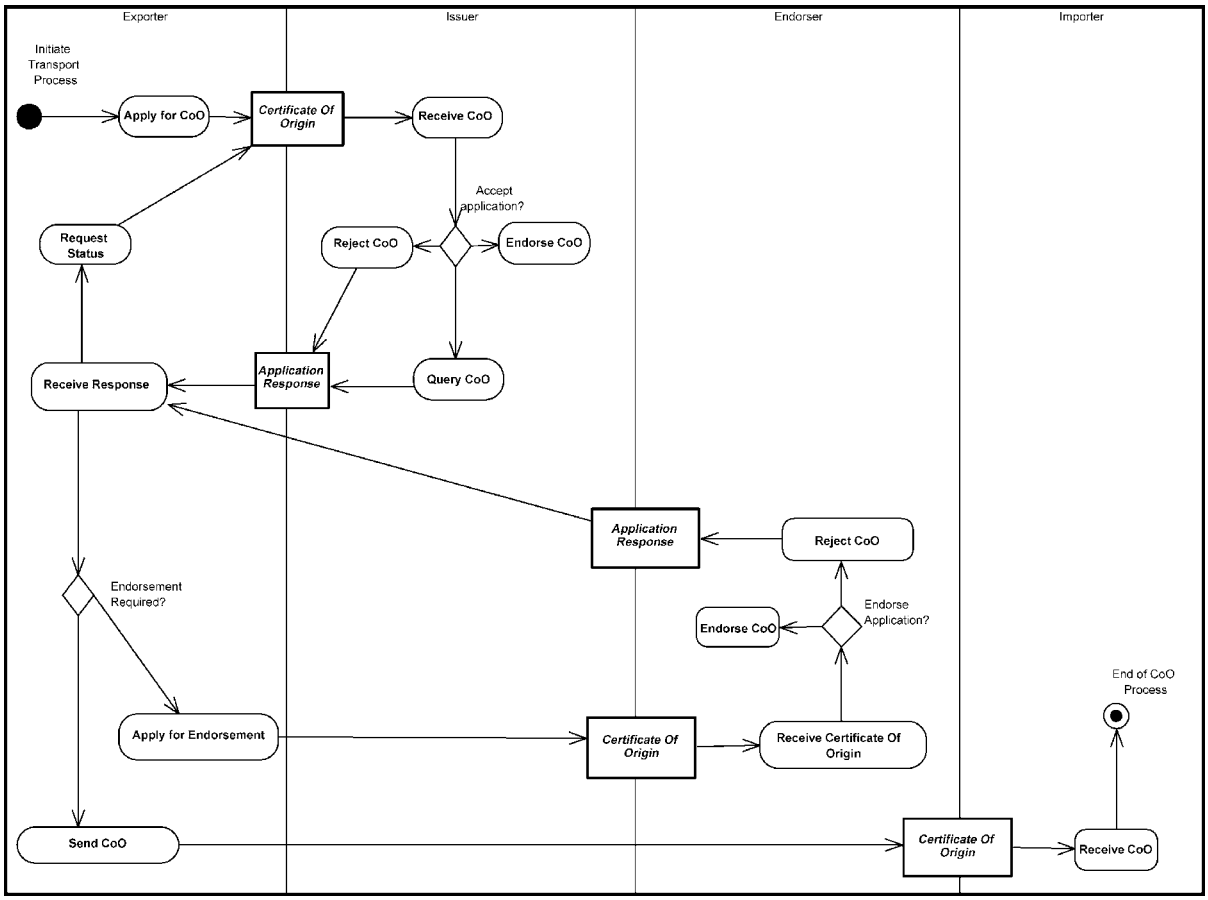


2.17 Certification of Origin of Goods

When a Consignor exports certain goods they may be required to attest to the origin of the goods. A Certificate of Origin is a document required by regulatory bodies declaring that goods in a particular international shipment are of a certain origin.

It is the responsibility of the Exporter to sign the Certificate of Origin Application document and submit it for authentication to a recognized authority (such as a local chamber of commerce or designated government agency or board). This party becomes the Endorser and will issue the Certificate of Origin document. To do this the Endorser must have access to other documents, such as the commercial Invoice and Bill of Lading, in order to verify the Exporter's claims that the goods originated in that country. In effect, the Certificate of Origin document is a dossier describing a set of related documents. After it is issued, the Certificate of Origin is sent to the Importer.

Figure 56. Certification of Origin of Goods Process



2.18 Intermodal Freight Management

Intermodal transport implies the use of a combination of transport modes. Any support for the management of such chains has to support the modal change of cargo flows from one mode to another in combination in order to create seamless sequences of transportation legs. Quite often the end legs are carried by road, but there are instances of short sea shipping, inland waterways, and rail being used as end legs.

The Intermodal Freight Management process differs from conventional freight management in that it is dependent on different transport modes. The focus is the multimodal transport chain as seen from the Transport User's point of view. The Transport User needs information about all the possible transport services that can be used to build a complete transport chain. If the choices to be made by the Transport User or his agent are based upon the qualities of the transport services themselves, and not by which transport mode is used, the description of the transport services, and the exchanges of information about the roles and services must be simple and common.

The roles of the various Parties are defined as follows:

- The *Transport User* is the role representing anyone who needs to have cargo transported. The Transport User provides the Transport Service Provider with instructions and detailed information about the cargo to be transported.
- The *Transport Service Provider* is the role that ensures the transport of the cargo from the origin to the destination. This includes the management of the transport services and the operation of the transport means and handling equipment. A Transport Service Provider may also provide administrative services required for moving the cargo, such as cargo inspection.
- The *Transportation Network Manager* is the role that extracts all information available regarding the infrastructure related to planning and executing transport and makes this information available to the Transport Service Provider.
- The *Transport Regulator* is the role that receives all mandatory reporting (and checks if reporting has been carried out) in order to ensure that all transport services are completed according to existing rules and regulations.

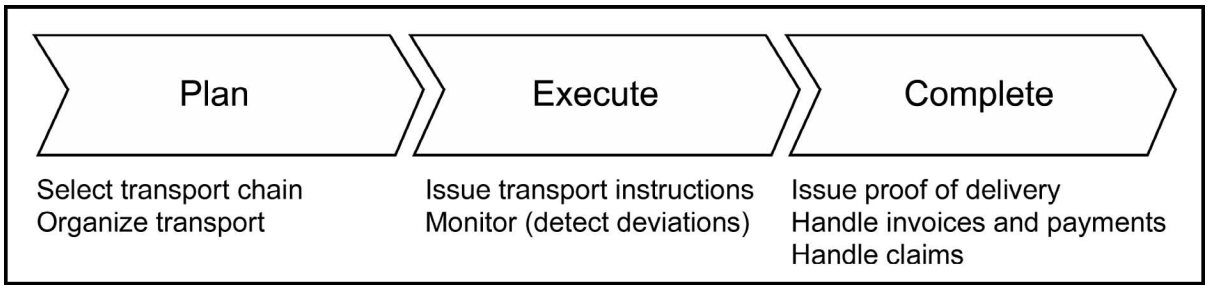
It should be noted that one person or organization may take on different roles. For example, a freight forwarder is, on the one hand, a Transport Service Provider when its client is a Transport User. On the other hand, the freight forwarder is a Transport User when it acquires services from subcontractors to ensure that a transport service is carried out between origin and destination. In so doing, the freight forwarder can operate as agent, thus arranging a contractual relationship between the carrier and the shipper, or as principal, thus organizing the transportation chain by concluding contracts in its own name on behalf of the shipper(s).

The Intermodal Freight Management process takes place in three stages:

- *Planning*: In this stage, the Transport Users express their transport demand in a standard format, the Transport Service Description Request. Transport Service Providers plan their transport services and announce them to Transport Users using the Transport Service Description. This stage also covers the arrangement of transport services between Transport Users and Transport Service Providers, establishing Transport Execution Plans. Once a Transport Execution Plan has been established, a Goods Item Itinerary is sent from the Transport Service Provider to the Transport User. The Goods Item Itinerary provides additional information related to the complete transport service.
- *Execution*: In this stage, Transport Service Providers perform the physical transport of the cargo, and they exchange information related to the status of the transported cargo with the Transport Users using the Transportation Status document. Furthermore, in this stage Transport Service Providers exchange regulatory information with Transport Regulators as well as receive status regarding the transport infrastructure from Transportation Network Managers using the Transport Progress Status document.

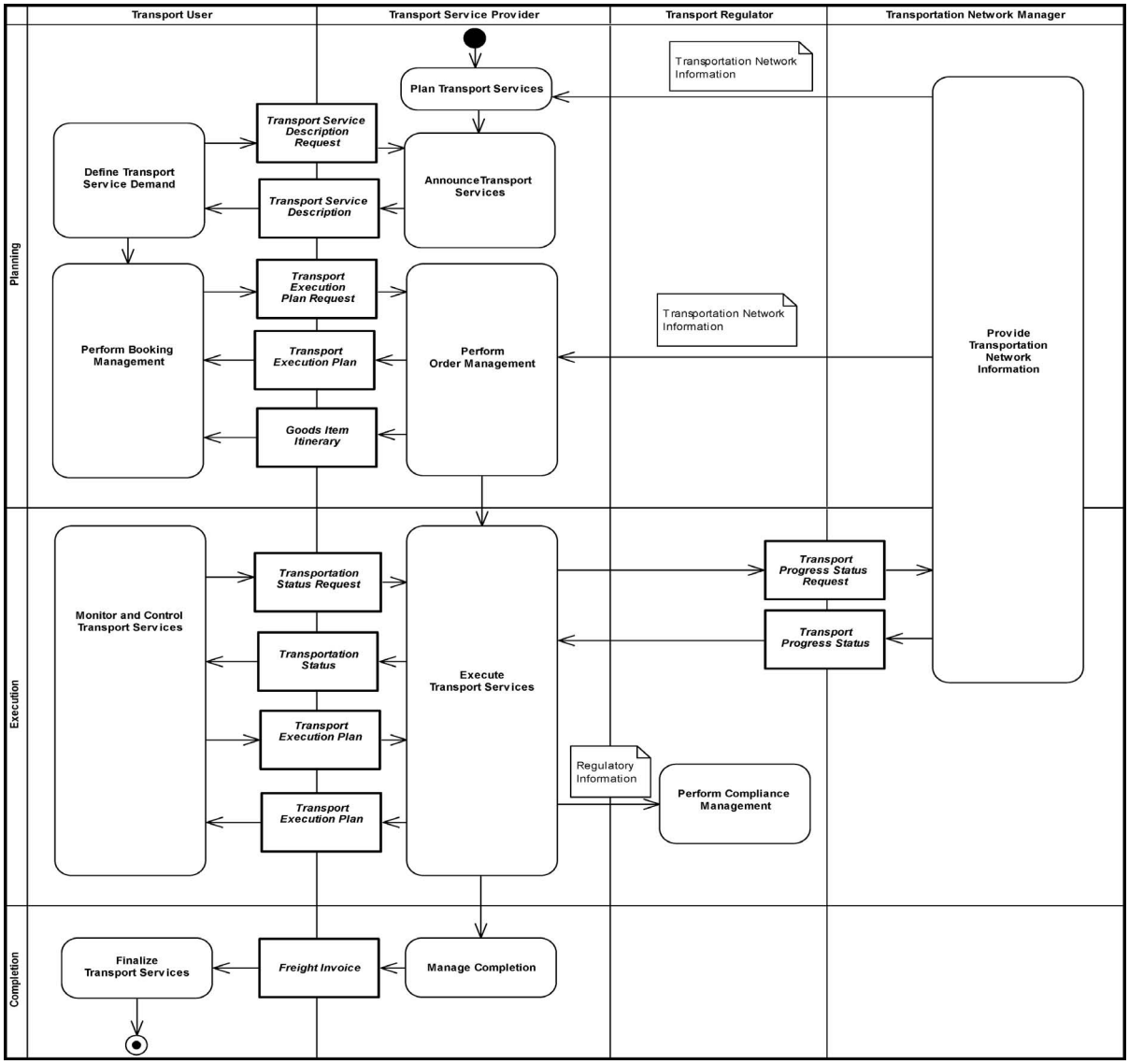
- *Completion*: This stage facilitates the issuing of proofs of delivery, claims, and invoices between Transport Service Providers and Transport Users.

Figure 57. The Generic Freight Management Process



These three stages are detailed in the following diagram, which shows the part played in the Intermodal Freight Management Process by the UBL document types Transport Service Description, Transport Service Description Request, Transport Execution Plan, Transport Execution Plan Request, Transportation Status, Transportation Status Request, Transport Progress Status, Transport Progress Status Request, Goods Item Itinerary, and Freight Invoice.

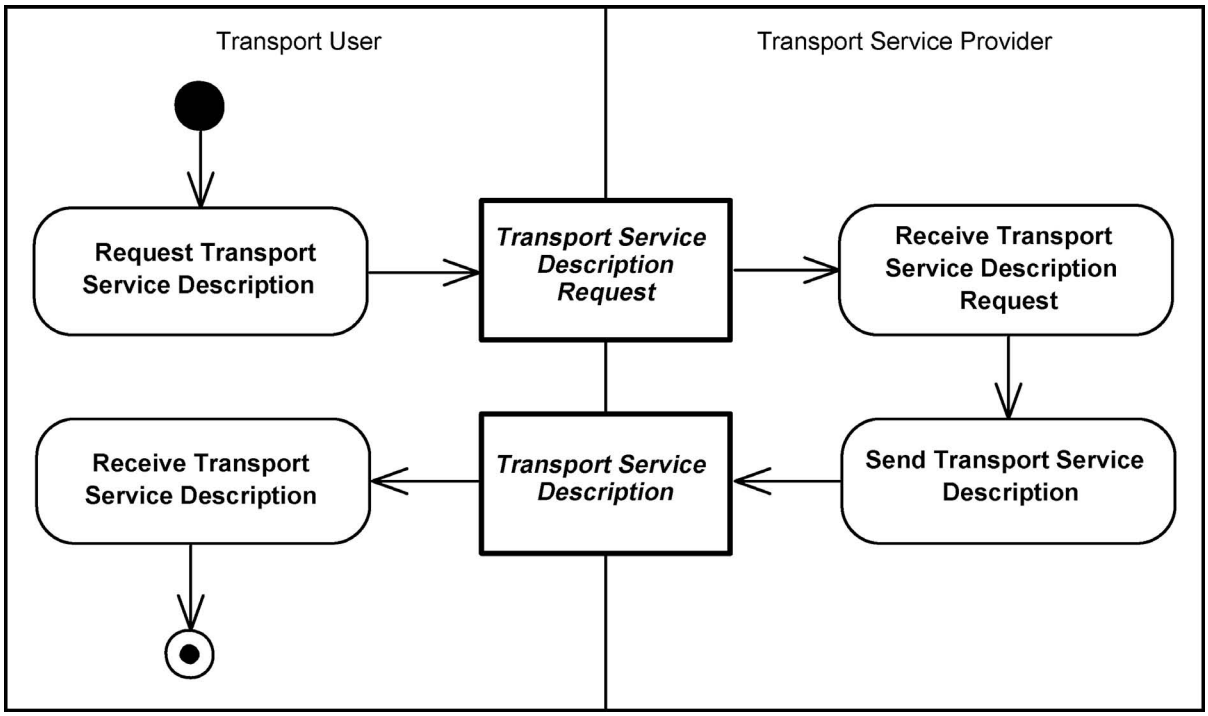
Figure 58. The Intermodal Freight Management Process



2.18.1 Announcing Intermodal Transport Services

The Transport Service Description is used to publish information about a transport service. A Transport Service Description Request is used to request such information. A transport service can be the physical transport of cargo between an origin and a destination, and it can also refer to other transport-related services such as terminal services, warehousing services, handling services, or document handling services.

Figure 59. Transport Service Description



2.18.2 Establishing a Transport Execution Plan

The Transport Execution Plan is a plan established between a Transport User and a Transport Service Provider in order to collaborate and document the details surrounding the provision of a required transport service. Depending on the nature of the transport service and the business relationship between the Transport User and the Transport Service Provider, the process of establishing a Transport Execution Plan may be carried out by means of multiple interactions between the two roles, from the initial request from the Transport User up to the final agreement of the Transport Execution Plan among the parties involved.

The following diagram (Figure 60, “Transport Execution Plan”) shows the message exchange involved in a basic scenario. A Transport Execution Plan Request is sent from the Transport User in order to request a transport service. If the Transport Service Provider accepts the transport service request, he responds with a confirmed Transport Execution Plan. If the Transport Service Provider does not accept the transport service request, he responds with a rejected Transport Execution Plan.

The handling of a Transport Service Request will in many cases depend upon whether or not there is a pre-established agreement between the Transport User and the Transport Service Provider. If there is a pre-established agreement, the Transport Service Request can typically be considered a call-off from the agreement between the two parties. (An established framework agreement or contract usually defines terms and conditions and a total capacity limit, e.g., 100 container spaces on a vessel per year. A call-off occurs when the Transport User places an order against this agreement, for example a booking of 10 of the 100 container spaces.) The Transport User can confirm the Transport Execution Plan Request without the need to make a careful examination of the Transport Execution Plan submitted by the Transport Service Provider. The Transport User then sends a Transport Execution Plan with a status

code indicating confirmation. Assuming acceptance by the Transport Service Provider, this scenario is considered a two-step choreography.

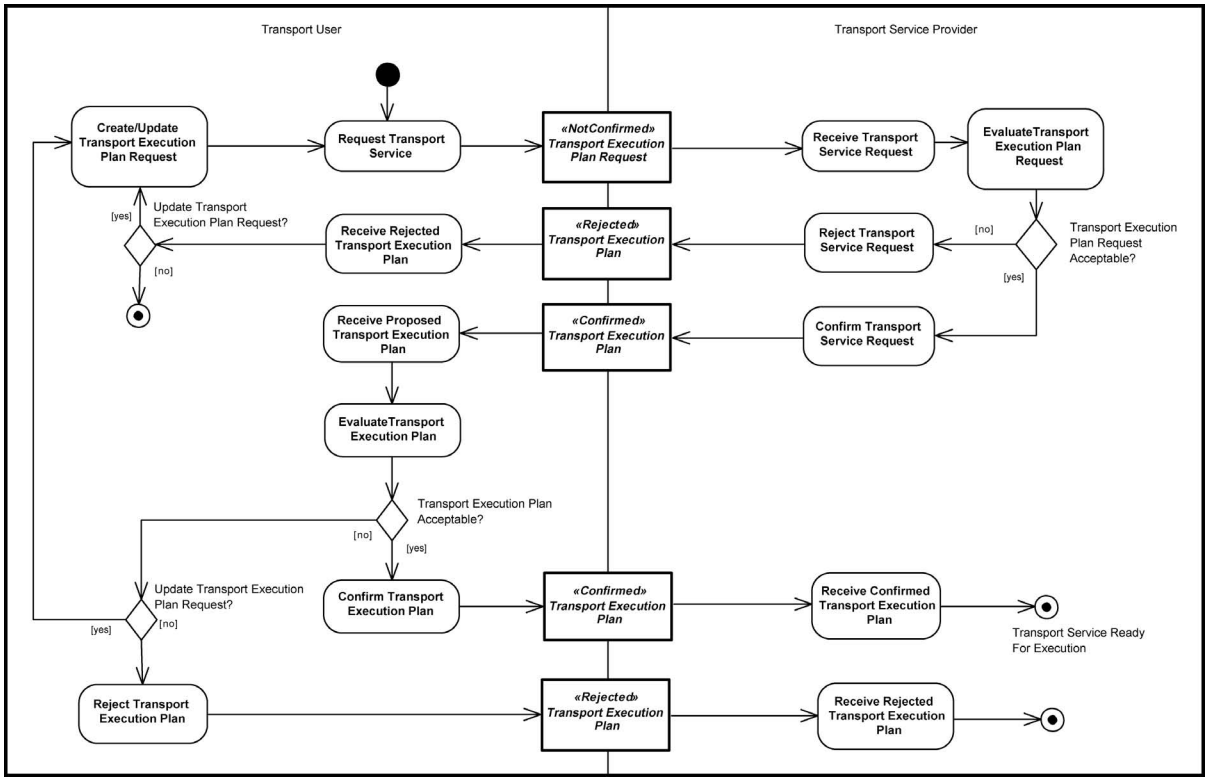
If a pre-established agreement does not exist (e.g., spot market services), the Transport User issues a Transport Execution Plan Request with a status code indicating that the Transport Execution Plan is not yet confirmed. The Transport User only confirms the Transport Execution Plan after a careful analysis of what has been submitted by the Transport Service Provider. This scenario is a three-step choreography where the Transport User confirms the Transport Execution Plan content in his second or subsequent response to the Transport Service Provider.

Updates to the Transport Execution Plan may be issued by either the Transport User or the Transport Service Provider. If the Transport User wants to update an existing Transport Execution Plan, a new instance of a Transport Execution Plan must be issued with reference to the original Transport Execution Plan. Similarly, if the Transport Service Provider wants to update an existing Transport Execution Plan, a new Transport Execution Plan replaces the original Transport Execution Plan with a reference to the original one. In either case, the Transport Execution Plan must include a document status code indicating that this is an update of the original content.

The cancellation of a Transport Execution Plan may be requested by either the Transport Service Provider or the Transport User. In either case, the Transport Execution Plan is sent with a document status code indicating that the Transport Execution Plan should be cancelled. For the cancellation to be effective, it must be accepted by the party receiving the cancellation request. Acceptance is signified by sending back a Transport Execution Plan with a status code indicating confirmation; rejection of the cancellation is signified by issuing a Transport Execution Plan with a status code indicating rejection.

Upon completion of the transport service covered by the Transport Execution Plan, a final Transport Execution Plan document is sent from the Transport Service Provider to the Transport User that includes a document status code indicating that the transport service is completed.

Figure 60. Transport Execution Plan

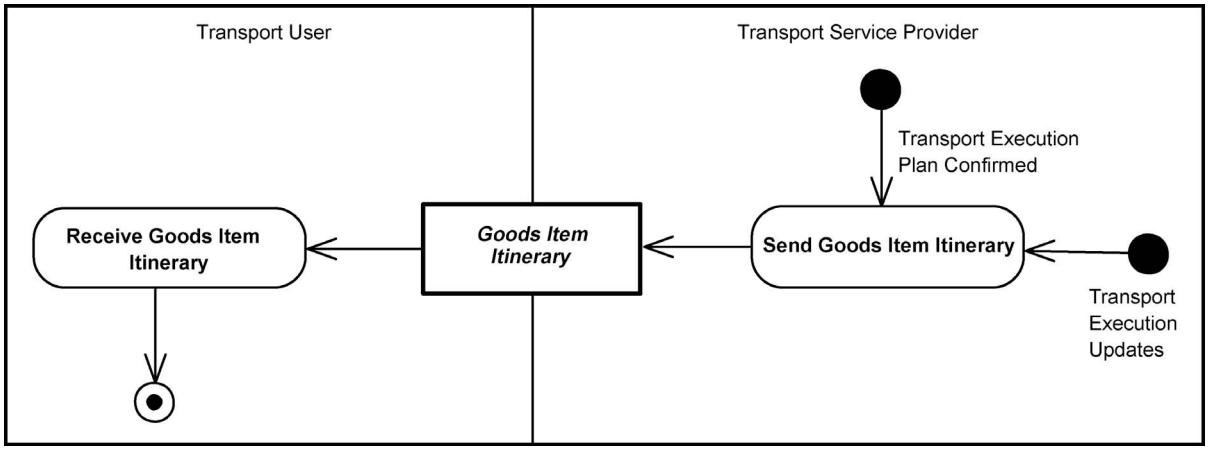


2.18.3 Providing an Itinerary for a Transport Service

The Goods Item Itinerary specifies the route and time schedule for one or more transported items and is issued from the Transport Service Provider to the Transport User. The Goods Item Itinerary is initially issued from the Transport Service Provider to the Transport User after a Transport Execution Plan is confirmed by both parties. It may contain one or more transport segments with different Transport Execution Plans employing different Transport Service Providers. One transport service (one Transport Execution Plan) may cover more than one segment (leg).

In addition to providing an overview of the initial route and time schedule, the Goods Item Itinerary is used to document progress by recording new estimated times for departure or arrival and actual departure and arrival times. So when updates to the initial transport execution schedule occur, a new version of the Goods Item Itinerary is issued to the Transport User. The Goods Item Itinerary thus contains information that may be used for analyzing the performance (in time) of transport services and for tracing the progress of cargo in transit if such analysis is required.

Figure 61. Goods Item Itinerary



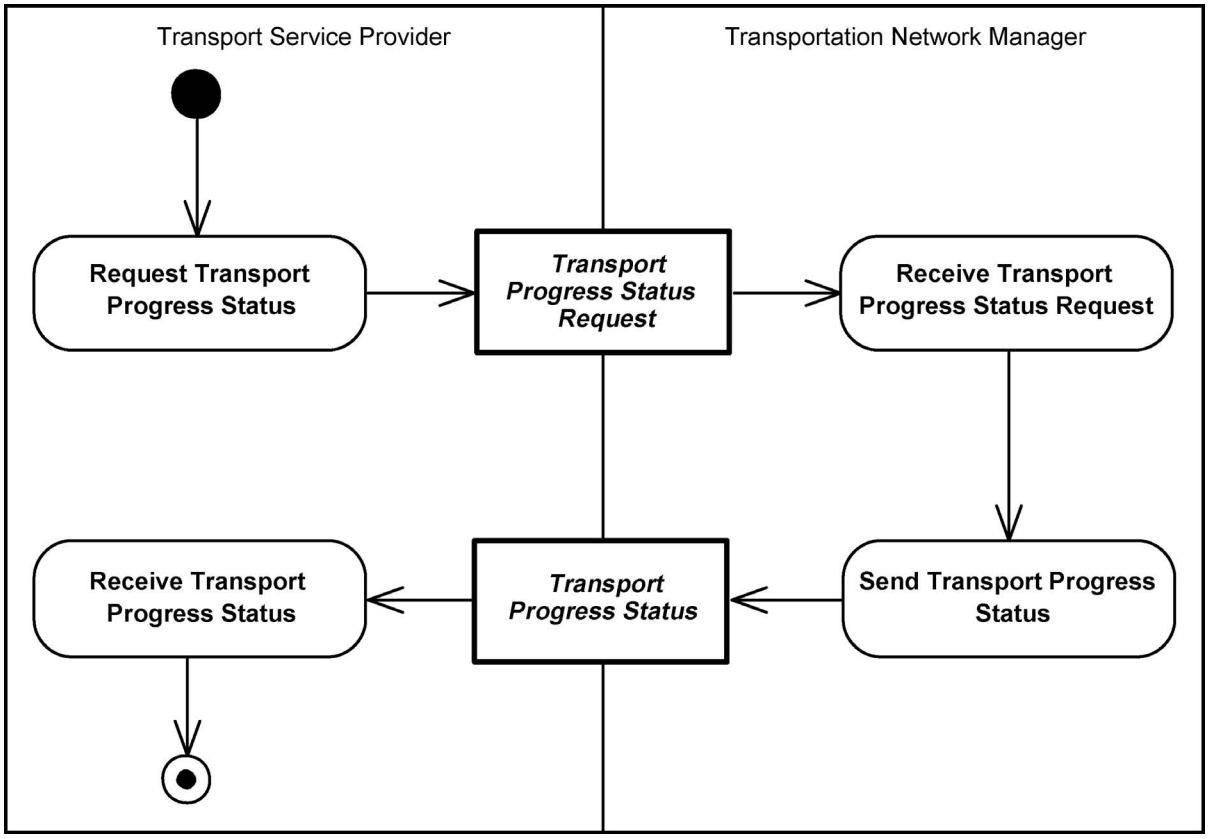
2.18.4 Reporting Transport Means Progress Status

The Transport Progress Status collects and reports information about the status of the transport means. The Transport Service Provider issues a Transport Progress Status Request to ask the Transportation Network Manager for status information related to a specific transport vehicle, using the vehicle identification number.

The Transportation Network Manager then provides information about the location and time schedule status to the Transport Service Provider. During a transport service, there might be a number of information providers taking on the Transportation Network Manager role, offering Transport Progress Statuses to the Transport Service Provider.

The most typical use of Transport Progress Status is to ask assistance from the Transportation Network Manager when estimated times of arrival are established. Reporting on the status of the goods themselves is covered by the Freight Status Reporting process (see Section 2.16, “Freight Status Reporting”).

Figure 62. Transport Progress Status



2.19 Party Roles

In the UBL supply chain processes, two main actors, Customer and Supplier, represent the key organizations or people involved in the processes. Each of these actors may play various roles. Some processes may also involve supplementary roles that may be provided by different parties.

The actual role undertaken is dependent on the context of use. For example, the Despatch Party and Delivery Party as applied to the Procurement process may differ in the Transportation process. In the Transportation Process, two of the main roles are the Transport User and the Transport Service Provider. The Transport User is the role responsible for purchasing a transport service, while the Transport Service Provider is the role responsible for selling and executing a transport service. Both the Customer and the Supplier may be responsible for purchasing and following up the transport of goods, hence both these actors may undertake the Transport User role. In other words, the role of a specific actor depends on the specific circumstances.

The following table contains a description of the typical roles for the actor known as Party. Note that some roles require an extension of the information entities required. In UBL 2.1, the following are roles that extend the Party structure: Customer Party, Supplier Party, Contracting Party, Endorser Party, and Qualifying Party.

Table 1. Party Roles

Actor	Role	Description	Example	Synonyms	Sends	Receives
Customer Party	Originator	The party that had the original demand for the goods and/or services and therefore initiated the procurement transaction. The Originator participates in pre-ordering activity either through Request for Quotation and Quotation or by receiving a Quotation as a response to a punchout transaction on a marketplace or Seller's website. If the Originator subsequently places an Order, the Originator adopts the role of Buyer. The Originator is typically the contact point for queries regarding the original requirement and may be referred to in an Order Change, Order Cancellation, or Order Response.	If an employee requests a computer, the employing company may become the Buyer, but the employee is the Originator. They need to receive information about the order.		Request for Quotation	Quotation
Customer Party	Buyer	The party that purchases the goods or services on behalf of the Originator. The Buyer may be referred to in Order Response, Despatch Advice, Fulfilment Cancellation, Invoice, Self Billed Invoice, Credit Note, and Statement.	A company may delegate the task of purchasing to a specialized group to consolidate orders and gain greater discounts.	Order Point	Order, Order Change, Order Cancellation, Fulfilment Cancellation	Order Response, Fulfilment Cancellation
Customer Party	Delivery	The party to whom goods should be delivered. The Delivery Party may be the same as the Originator. The Delivery Party must be referred to at line item level in Request for Quotation, Quotation, Order, Order Change, Order Cancellation, and Order Response. The Delivery Party may be referred to at line level in Invoice, Self Billed Invoice, Credit Note, and Debit Note. The Delivery Party may be stipulated in a transport contract.	If a municipality buys a wheelchair for a citizen, the wheelchair must be delivered to the citizen (the Delivery Party). In such cases the citizen may be notified before delivering the wheelchair.	Delivery Point, Destination Party, Receiver, Recipient	Receipt Advice	Despatch Advice
Customer Party	Accounting Customer	The party responsible for making settlement relating to a purchase and resolving billing issues using a Debit Note. The Accounting Customer must be referred to in an Order and may be referred to in an Order Response. In a Self Billing scenario, the Accounting Customer is responsible for calculating and issuing tax invoices.	If a kindergarten buys some toys they may be the Originator, Buyer, and Delivery Party, but the municipality may play the role of Accounting Customer—they are going to pay for it.	Invoicee, Accounts Payable, Debtor	In a traditional Billing scenario: Debit Note, Account Response, and Remittance Advice In a Self Billing scenario: Self Billed Invoice, Self Billed Credit Note, and Remittance Advice	In a traditional Billing scenario: Invoice, Credit Note, and Statement; in a Self Billing scenario: Credit Note, Account Response, and Statement

Actor	Role	Description	Example	Synonyms	Sends	Receives
Supplier Party	Seller	The party responsible for handling Originator and Buyer services. The Seller party is legally responsible for providing the goods to the Buyer. The Seller party receives and quotes against Requests for Quotation and may provide information to the Buyer's requisitioning process through Catalogues and Quotations.	The organization that sells wheelchairs to municipalities.	Sales Point, Provider, Customer Manager	Quotation, Order Response, Order Response Simple, Catalogue, Catalogue Deletion, Catalogue Item Specification Update, Catalogue Pricing Update, Fulfilment Cancellation	Request for Quotation, Order, Order Change, Order Cancellation, Request for Catalogue, Fulfilment Cancellation
Supplier Party	Despatch	The party where goods are to be collected from. The Despatch Party may be stipulated in a transport contract.	The wheelchair Supplier may store chairs at a local warehouse. The warehouse will actually despatch the chair to the Delivery Party. The local warehouse is then the Despatch Party.	Despatch Point, Shipper, Sender	Despatch Advice	Receipt Advice
Supplier Party	Accounting Supplier	The party who claims the payment and is responsible for resolving billing issues and arranging settlement.	There are cases where the Accounting Supplier is not the Seller party. For example, factoring, where the invoicing is outsourced to another company.	Accounts Receivable, Invoice Issuer, Creditor	In a traditional Billing scenario: Invoice, Credit Note, and Statement; in a Self Billing scenario: Credit Note, Account Response, and Statement	In a traditional Billing scenario: Debit Note, Account Response, and Remittance Advice In a Self Billing scenario: Self Billed Invoice, Self Billed Credit Note, and Remittance Advice
Supplier Party	Payee	The party to whom the Invoice is paid.	The Accounting Supplier may not be the party to be paid due to changes in the organization, e.g., a company merger.	Accounts Receivable, Creditor		Remittance Advice
Customer Party	Contractor	The party responsible for the contract to which the Catalogue relates.	An organization has a central office for maintaining catalogues of approved items for purchase.	Central Catalogue Party, Purchasing Manager	Request for Catalogue	Catalogue, Catalogue Deletion, Catalogue Item Specification Update, Catalogue Pricing Update
Party	Provider	The party responsible for the integrity of the information provided about an item.	The manufacturer may publish and maintain the data sheets about a product.		Catalogue, Catalogue Deletion, Catalogue Item Specification Update, Catalogue Pricing Update	

Actor	Role	Description	Example	Synonyms	Sends	Receives
Party	Receiver	A general role, describing the receiver of a document. For a catalogue, this can be the customer, a potential customer, or a third party exposing the document, for instance, an interim broker.	A marketplace may receive an Application Response.			Catalogue, Catalogue Deletion, Catalogue Item Specification Update, Catalogue Pricing Update, Application Response
Party	Sender	The party sending a document.	A marketplace may send an Application Response.		Application Response	
Party	Consignor	The party consigning the goods as stipulated in the transport contract. A Buyer, Delivery, Seller, or Despatcher Party may also play the role of Consignor. Also known as the Transport User. The Consignor may be stipulated in a transport contract.	The wheelchair Supplier may source from a local warehouse. The Freight Forwarder will collect the chair from the local warehouse, which is thus the Consignor. In this case, the warehouse also plays the role of Despatch Party to the Freight Forwarder.	Despatch Point, Shipper, Sender, Transport User	Forwarding Instructions, Packing List	Bill of Lading, Waybill, Freight Invoice, Transportation Status
Party	Consignee	The party receiving a consignment of goods as stipulated in the transport contract.	The party taking responsibility for the receipt of the consignment covering the wheelchair.	Delivery Point, Transport Service Buyer	Forwarding Instructions, Freight Invoice	Bill of Lading, Waybill, Freight Invoice, Transportation Status
Party	Freight Forwarder	The party arranging the carriage of goods, including connected services and/or associated formalities, on behalf of a Consignor or Consignee. Also known as the Transport Service Provider. The Freight Forwarder may also be the Carrier. The Freight Forwarder may create an Invoice and bill to the Transport Service Buyer for the transportation service provided.	The Consignor may have a contract with this Freight Forwarder, which is a Transport Services Provider, to arrange all their transport needs.	Shipping Agent, Broker, Courier, Transport Service Provider	Forwarding Instructions, Freight Invoice, Transportation Status	Bill of Lading, Waybill, Packing List
Party	Carrier	The party providing physical transport services.	The Freight Forwarder may engage an airline company to deliver the wheelchair. The airline is then the Carrier and delivers the chair to the Delivery Party.	Freight Haulier, Shipper, Ships Agent, Shipping Company, Airline, Rail Operator, Road Haulier	Bill of Lading, Waybill	Forwarding Instructions

Actor	Role	Description	Example	Synonyms	Sends	Receives
Party	Exporter	The party who makes regulatory export declarations, or on whose behalf regulatory export declarations are made, and who is the owner of the goods or has similar right of disposal over them at the time when the declaration is accepted.	The wheelchair Supplier has to apply for a Certificate of Origin in order to sell the chairs overseas.	Seller, Consignor	Certificate of Origin	Application Response
Party	Endorser	The party appointed by the Government of a country who has the right to certify a Certificate of Origin. This endorsement restricts goods imported from certain countries for political or other reasons.	The Government agency validates all the information provided by Exporter for Certificate of Origin approval.	Authorized Organization, Embassy	Certificate of Origin, Application Response	Certificate of Origin
Party	Importer	The party who makes, or on whose behalf an agent or other authorized person makes, an import declaration. This may include a person who has possession of the goods or to whom the goods are consigned.	A specialized group in a company consolidates the purchase request and handles the receiving of goods.	Order Point, Delivery Party, Buyer, Customer, Consignee		Certificate of Origin
Party	Transport User	The Transport User is the role representing anyone who has a demand for transport services, books transport services, and follows up the execution of such services.	The manufacturer has to order transport of products from a carrier or freight forwarder (Transport Service Provider).	Transport Buyer, Logistics Service Client	Transport Execution Plan Request, Transportation Status Request, Transport Service Description Request	Transport Execution Plan, Transportation Status, Transport Service Description, Goods Item Itinerary
Party	Transport Service Provider	The Transport Service Provider is the role that plans, markets and performs transport services.	The carrier or a freight forwarder who arranges for transport services on behalf of a manufacturer (Transport User)	Transport Provider, Transport Seller, Logistics Service Provider	Transport Execution Plan, Transportation Status, Transport Service Description, Transport Progress Status Request, Goods Item Itinerary	Transport Execution Plan Request, Transportation Status Request, Transport Service Description Request, Transport Progress Status
Party	Transportation Network Manager	The Transportation Network Manager is the role that extracts all information available regarding the infrastructure (static/dynamic) related to planning and executing transport and makes this information available to the Transport Service Provider. During a transport service, or even during a single leg, the Transport Service Provider may rely on information from several Transportation Network Managers.	The Traffic Information Centre (TIC) issuing information related to road work and/or traffic conditions as a service to a Transport Service Provider	Road Administration, Traffic Information Centre, Coastal Administration, Harbor Master, Railway Administration, Infrastructure Manager	Transport Progress Status	Transport Progress Status Request

3 UBL 2.1 Schemas

The UBL 2.1 XSD schemas are the only normative representations of the UBL 2.1 document types and library components.

All of the UBL 2.1 XSD schemas are contained in the `xsd` subdirectory of the UBL 2.1 release package (see Appendix A, *Release Notes (Non-Normative)* for more information regarding the structure of the 2.1 release package and Section 3.3, "Schema Dependencies" for information regarding dependencies among the schema modules). The `xsd` directory is further subdivided into an `xsd/maindoc` subdirectory containing the schemas for individual 2.1 document types and an `xsd/common` subdirectory containing schemas in the UBL common library. For convenience in implementing the schemas, parallel (and technically non-normative) "runtime" sets with the annotation elements stripped out are provided in the `xsdrt` directory.

3.1 UBL 2.1 Document Schemas

The tables that follow describe each of the UBL 2.1 document types. Along with a link to the normative schema for each document type, each table provides links to the corresponding "runtime" schema, model spreadsheets (see Appendix C, *The UBL 2.1 Data Model (Non-Normative)*), summary report (see Section C.5, "Summary Reports"), RELAX NG schema (see Section H.2, "UBL 2.1 RELAX NG Schemas"), and example instance, if any (see Appendix G, *UBL 2.1 Example Document Instances (Non-Normative)*).

3.1.1 Application Response

Description: A document to indicate the application's response to a transaction. This may be a business response and/or a technical response, sent automatically by an application or initiated by a user.

Processes involved	Any
Submitter role	Sender
Receiver role	Receiver
Normative schema	`xsd/maindoc/UBL-ApplicationResponse-2.1.xsd`
Runtime schema	`xsdrt/maindoc/UBL-ApplicationResponse-2.1.xsd`
RELAX NG schema	`rnc/versions/UBL-ApplicationResponse-2.1.rnc`
Document model (ODF)	`mod/maindoc/UBL-ApplicationResponse-2.1.ods`
Document model (Excel)	`mod/maindoc/UBL-ApplicationResponse-2.1.xls`
Summary report	`mod/summary/reports/UBL-ApplicationResponse-2.1.html`

3.1.2 Attached Document

Description: A UBL wrapper that allows a document of any kind to be packaged with the UBL document that references it.

Processes involved	Any
Submitter role	Sender
Receiver role	Receiver
Normative schema	`xsd/maindoc/UBL-AttachedDocument-2.1.xsd`
Runtime schema	`xsdrt/maindoc/UBL-AttachedDocument-2.1.xsd`
RELAX NG schema	`rnc/versions/UBL-AttachedDocument-2.1.rnc`
Document model (ODF)	`mod/maindoc/UBL-AttachedDocument-2.1.ods`

Document model (Excel)	mod/maindoc/UBL-AttachedDocument-2.1.xls
Summary report	mod/summary/reports/UBL-AttachedDocument-2.1.html

3.1.3 Awarded Notification

Description: The document used to communicate the contract award to the winner

Processes involved	Tendering
Submitter role	Contracting Authority
Receiver role	Tenderer
Normative schema	xsd/maindoc/UBL-AwardedNotification-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-AwardedNotification-2.1.xsd
RELAX NG schema	rnc/versions/UBL-AwardedNotification-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-AwardedNotification-2.1.ods
Document model (Excel)	mod/maindoc/UBL-AwardedNotification-2.1.xls
Summary report	mod/summary/reports/UBL-AwardedNotification-2.1.html

3.1.4 Bill of Lading

Description: The Bill of Lading conveys information about an instance of a transportation service and may under some circumstances serve as a contractual document for the service. See Section 2.15.3, “Bill of Lading” and compare with Section 2.15.4, “Waybill”.

Processes involved	Freight Management
Submitter role	Freight Forwarder, Carrier
Receiver role	Consignor (or Consignee), Freight Forwarder
Normative schema	xsd/maindoc/UBL-BillOfLading-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-BillOfLading-2.1.xsd
RELAX NG schema	rnc/versions/UBL-BillOfLading-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-BillOfLading-2.1.ods
Document model (Excel)	mod/maindoc/UBL-BillOfLading-2.1.xls
Summary report	mod/summary/reports/UBL-BillOfLading-2.1.html

3.1.5 Call for Tenders

Description: The document used for a Contracting Party to define the procurement project to buy goods, services or works during an specified period.

Processes involved	Tendering
Submitter role	Contracting Authority
Receiver role	Tenderer
Normative schema	xsd/maindoc/UBL-CallForTenders-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-CallForTenders-2.1.xsd
RELAX NG schema	rnc/versions/UBL-CallForTenders-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-CallForTenders-2.1.ods
Document model (Excel)	mod/maindoc/UBL-CallForTenders-2.1.xls

Summary report	mod/summary/reports/UBL-CallForTenders-2.1.html

3.1.6 Catalogue

Description: The document that describes items, prices, and price validity.

Processes involved	Catalogue, Create Catalogue, Delete Catalogue, Update Catalogue Item Specification, Update Catalogue Pricing; Initial Stocking of the Area by Producer, Permanent Replenishment, Price Adjustments, Transfer of Base Item Catalogue, Changes to the Item Catalogue, Changes to the Article Catalogue
Submitter role	Seller
Receiver role	Contracting Party
Normative schema	xsd/maindoc/UBL-Catalogue-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-Catalogue-2.1.xsd
RELAX NG schema	rnc/versions/UBL-Catalogue-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-Catalogue-2.1.ods
Document model (Excel)	mod/maindoc/UBL-Catalogue-2.1.xls
Summary report	mod/summary/reports/UBL-Catalogue-2.1.html

3.1.7 Catalogue Deletion

Description: The document used to cancel an entire Catalogue.

Processes involved	Catalogue
Submitter role	Seller
Receiver role	Contracting Party
Normative schema	xsd/maindoc/UBL-CatalogueDeletion-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-CatalogueDeletion-2.1.xsd
RELAX NG schema	rnc/versions/UBL-CatalogueDeletion-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-CatalogueDeletion-2.1.ods
Document model (Excel)	mod/maindoc/UBL-CatalogueDeletion-2.1.xls
Summary report	mod/summary/reports/UBL-CatalogueDeletion-2.1.html

3.1.8 Catalogue Item Specification Update

Description: The document used to update information about Items (e.g., technical descriptions and properties) on an existing Catalogue.

Processes involved	Catalogue
Submitter role	Seller
Receiver role	Contracting Party
Normative schema	xsd/maindoc/UBL-CatalogueItemSpecificationUpdate-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-CatalogueItemSpecificationUpdate-2.1.xsd
RELAX NG schema	rnc/versions/UBL-CatalogueItemSpecificationUpdate-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-CatalogueItemSpecificationUpdate-2.1.ods
Document model (Excel)	mod/maindoc/UBL-CatalogueItemSpecificationUpdate-2.1.xls

Summary report	mod/summary/reports/UBL-CatalogueItemSpecificationUpdate-2.1.html

3.1.9 Catalogue Pricing Update

Description: The document used to update information about prices on an existing Catalogue.

Processes involved	Catalogue
Submitter role	Seller
Receiver role	Contracting Party
Normative schema	xsd/maindoc/UBL-CataloguePricingUpdate-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-CataloguePricingUpdate-2.1.xsd
RELAX NG schema	rnc/versions/UBL-CataloguePricingUpdate-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-CataloguePricingUpdate-2.1.ods
Document model (Excel)	mod/maindoc/UBL-CataloguePricingUpdate-2.1.xls
Summary report	mod/summary/reports/UBL-CataloguePricingUpdate-2.1.html

3.1.10 Catalogue Request

Description: The document used to request a Catalogue.

Processes involved	Catalogue
Submitter role	Contracting Party
Receiver role	Seller
Normative schema	xsd/maindoc/UBL-CatalogueRequest-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-CatalogueRequest-2.1.xsd
RELAX NG schema	rnc/versions/UBL-CatalogueRequest-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-CatalogueRequest-2.1.ods
Document model (Excel)	mod/maindoc/UBL-CatalogueRequest-2.1.xls
Summary report	mod/summary/reports/UBL-CatalogueRequest-2.1.html

3.1.11 Certificate of Origin

Description: A document that describes the Certificate of Origin.

Processes involved	Certification of Origin of Goods
Submitter role	Exporter, Issuer
Receiver role	Issuer, Importer
Normative schema	xsd/maindoc/UBL-CertificateOfOrigin-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-CertificateOfOrigin-2.1.xsd
RELAX NG schema	rnc/versions/UBL-CertificateOfOrigin-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-CertificateOfOrigin-2.1.ods
Document model (Excel)	mod/maindoc/UBL-CertificateOfOrigin-2.1.xls
Summary report	mod/summary/reports/UBL-CertificateOfOrigin-2.1.html

3.1.12 Contract Award Notice

Description: The document published by a Contracting Party to announce the awarding of a procurement project.

Processes involved	Tendering
Submitter role	Contracting Authority
Receiver role	Tenderer
Normative schema	xsd/maindoc/UBL-ContractAwardNotice-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-ContractAwardNotice-2.1.xsd
RELAX NG schema	rnc/versions/UBL-ContractAwardNotice-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-ContractAwardNotice-2.1.ods
Document model (Excel)	mod/maindoc/UBL-ContractAwardNotice-2.1.xls
Summary report	mod/summary/reports/UBL-ContractAwardNotice-2.1.html

3.1.13 Contract Notice

Description: The document used for a Contracting Party to announce the project to buy goods, services or works.

Processes involved	Tendering
Submitter role	Contracting Authority
Receiver role	Tenderer
Normative schema	xsd/maindoc/UBL-ContractNotice-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-ContractNotice-2.1.xsd
RELAX NG schema	rnc/versions/UBL-ContractNotice-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-ContractNotice-2.1.ods
Document model (Excel)	mod/maindoc/UBL-ContractNotice-2.1.xls
Summary report	mod/summary/reports/UBL-ContractNotice-2.1.html

3.1.14 Credit Note

Description: The document used to specify credits due to the Debtor from the Creditor.

Processes involved	Billing
Submitter role	Supplier Accounting Party
Receiver role	Customer Accounting Party
Normative schema	xsd/maindoc/UBL-CreditNote-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-CreditNote-2.1.xsd
RELAX NG schema	rnc/versions/UBL-CreditNote-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-CreditNote-2.1.ods
Document model (Excel)	mod/maindoc/UBL-CreditNote-2.1.xls
Summary report	mod/summary/reports/UBL-CreditNote-2.1.html
UBL 2.0 example instance	xml/UBL-CreditNote-2.0-Example.xml
UBL 2.1 example instance	xml/UBL-CreditNote-2.1-Example.xml

3.1.15 Debit Note

Description: The document used to specify debits made by the Debtor.

Processes involved	Billing
Submitter role	Customer Accounting Party
Receiver role	Supplier Accounting Party
Normative schema	xsd/maindoc/UBL-DebitNote-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-DebitNote-2.1.xsd
RELAX NG schema	rnc/versions/UBL-DebitNote-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-DebitNote-2.1.ods
Document model (Excel)	mod/maindoc/UBL-DebitNote-2.1.xls
Summary report	mod/summary/reports/UBL-DebitNote-2.1.html
UBL 2.1 example instance	xml/UBL-DebitNote-2.1-Example.xml

3.1.16 Despatch Advice

Description: The document used to describe the despatch or delivery of goods and services.

Processes involved	Fulfilment
Submitter role	Despatch
Receiver role	Delivery
Normative schema	xsd/maindoc/UBL-DespatchAdvice-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-DespatchAdvice-2.1.xsd
RELAX NG schema	rnc/versions/UBL-DespatchAdvice-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-DespatchAdvice-2.1.ods
Document model (Excel)	mod/maindoc/UBL-DespatchAdvice-2.1.xls
Summary report	mod/summary/reports/UBL-DespatchAdvice-2.1.html
UBL 2.0 example instance	xml/UBL-DespatchAdvice-2.0-Example.xml

3.1.17 Despatch Cancellation

****** NEW DOCTYPE ****** description etc. needed from PSC

3.1.18 Document Status

Description: A document used to provide information about document status.

Processes involved	Any collaboration
Submitter role	Party currently controlling Status of the collaboration
Receiver role	Party requesting Status on collaboration
Normative schema	xsd/maindoc/UBL-DocumentStatus-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-DocumentStatus-2.1.xsd
RELAX NG schema	rnc/versions/UBL-DocumentStatus-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-DocumentStatus-2.1.ods
Document model (Excel)	mod/maindoc/UBL-DocumentStatus-2.1.xls

Summary report	mod/summary/reports/UBL-DocumentStatus-2.1.html

3.1.19 Document Status Request

Description: A document used to request the status of another document.

Processes involved	Any collaboration
Submitter role	Party requesting Status on collaboration
Receiver role	Party currently controlling Status of the collaboration
Normative schema	xsd/maindoc/UBL-DocumentStatusRequest-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-DocumentStatusRequest-2.1.xsd
RELAX NG schema	rnc/versions/UBL-DocumentStatusRequest-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-DocumentStatusRequest-2.1.ods
Document model (Excel)	mod/maindoc/UBL-DocumentStatusRequest-2.1.xls
Summary report	mod/summary/reports/UBL-DocumentStatusRequest-2.1.html

3.1.20 Exception Criteria

Description: Used to specify basic information about the content of the message including version number, creation date and time.

Processes involved	Collaborative Planning, Forecasting and Replenishment
Submitter role	Buyer, Seller
Receiver role	Buyer, Seller
Normative schema	xsd/maindoc/UBL-ExceptionCriteria-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-ExceptionCriteria-2.1.xsd
RELAX NG schema	rnc/versions/UBL-ExceptionCriteria-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-ExceptionCriteria-2.1.ods
Document model (Excel)	mod/maindoc/UBL-ExceptionCriteria-2.1.xls
Summary report	mod/summary/reports/UBL-ExceptionCriteria-2.1.html
UBL 2.1 example instance	xml/UBL-ExceptionCriteria-2.1-Example.xml

3.1.21 Exception Notification

Description: The document used to notify an exception.

Processes involved	Collaborative Planning, Forecasting and Replenishment
Submitter role	Buyer, Seller
Receiver role	Buyer, Seller
Normative schema	xsd/maindoc/UBL-ExceptionNotification-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-ExceptionNotification-2.1.xsd
RELAX NG schema	rnc/versions/UBL-ExceptionNotification-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-ExceptionNotification-2.1.ods
Document model (Excel)	mod/maindoc/UBL-ExceptionNotification-2.1.xls
Summary report	mod/summary/reports/UBL-ExceptionNotification-2.1.html
UBL 2.1 example instance	xml/UBL-ExceptionNotification-2.1-Example.xml

3.1.22 Forecast

Description: The document used to specify a forecast.

Processes involved	Collaborative Planning, Forecasting and Replenishment
Submitter role	Buyer, Seller
Receiver role	Buyer, Seller
Normative schema	xsd/maindoc/UBL-Forecast-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-Forecast-2.1.xsd
RELAX NG schema	rnc/versions/UBL-Forecast-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-Forecast-2.1.ods
Document model (Excel)	mod/maindoc/UBL-Forecast-2.1.xls
Summary report	mod/summary/reports/UBL-Forecast-2.1.html
UBL 2.1 example instance	xml/UBL-Forecast-2.1-Example.xml

3.1.23 Forecast Revision

Description: The document used to revise a Forecast.

Processes involved	Collaborative Planning, Forecasting and Replenishment
Submitter role	Buyer, Seller
Receiver role	Buyer, Seller
Normative schema	xsd/maindoc/UBL-ForecastRevision-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-ForecastRevision-2.1.xsd
RELAX NG schema	rnc/versions/UBL-ForecastRevision-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-ForecastRevision-2.1.ods
Document model (Excel)	mod/maindoc/UBL-ForecastRevision-2.1.xls
Summary report	mod/summary/reports/UBL-ForecastRevision-2.1.html
UBL 2.1 example instance	xml/UBL-ForecastRevision-2.1-Example.xml

3.1.24 Forwarding Instructions

Description: The document issued to a forwarder, giving instructions regarding the action to be taken for the forwarding of goods described therein. See Section 2.15.1, “Forwarding Instructions”.

Processes involved	Freight Management
Submitter role	Consignor (or Consignee), Freight Forwarder
Receiver role	Freight Forwarder, Carrier
Normative schema	xsd/maindoc/UBL-ForwardingInstructions-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-ForwardingInstructions-2.1.xsd
RELAX NG schema	rnc/versions/UBL-ForwardingInstructions-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-ForwardingInstructions-2.1.ods
Document model (Excel)	mod/maindoc/UBL-ForwardingInstructions-2.1.xls
Summary report	mod/summary/reports/UBL-ForwardingInstructions-2.1.html
UBL 2.0 example instance	xml/UBL-ForwardingInstructions-2.0-Example-International.xml

3.1.25 Freight Invoice

Description: A document stating the charges incurred for the logistics service.

Processes involved	Freight Billing
Submitter role	Freight Forwarder
Receiver role	Consignor or Consignee
Normative schema	xsd/maindoc/UBL-FreightInvoice-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-FreightInvoice-2.1.xsd
RELAX NG schema	rnc/versions/UBL-FreightInvoice-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-FreightInvoice-2.1.ods
Document model (Excel)	mod/maindoc/UBL-FreightInvoice-2.1.xls
Summary report	mod/summary/reports/UBL-FreightInvoice-2.1.html
UBL 2.1 example instance	xml/UBL-FreightInvoice-2.1-Example.xml

3.1.26 Fulfilment Cancellation

Description: A document used to cancel an entire Despatch Advice or Receipt Advice.

Processes involved	Fulfilment with Despatch Advice, Fulfilment with Receipt Advice
Submitter role	*****
Receiver role	*****
Normative schema	xsd/maindoc/UBL-FulfilmentCancellation-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-FulfilmentCancellation-2.1.xsd
RELAX NG schema	rnc/versions/UBL-FulfilmentCancellation-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-FulfilmentCancellation-2.1.ods
Document model (Excel)	mod/maindoc/UBL-FulfilmentCancellation-2.1.xls
Summary report	mod/summary/reports/UBL-FulfilmentCancellation-2.1.html
UBL 2.1 example instance	xml/UBL-FulfilmentCancellation-2.1-Example.xml

3.1.27 Goods Item Itinerary

Description: A document specifying the route and the time schedule for a transport of a Goods Item for all segments in a transport service.

Processes involved	Intermodal Freight Management
Submitter role	Transport Service Provider
Receiver role	Transport User
Normative schema	xsd/maindoc/UBL-GoodsItemItinerary-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-GoodsItemItinerary-2.1.xsd
RELAX NG schema	rnc/versions/UBL-GoodsItemItinerary-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-GoodsItemItinerary-2.1.ods
Document model (Excel)	mod/maindoc/UBL-GoodsItemItinerary-2.1.xls
Summary report	mod/summary/reports/UBL-GoodsItemItinerary-2.1.html
UBL 2.1 example instance	xml/UBL-GoodsItemItinerary-2.1-Example.xml

3.1.28 Guarantee Certificate

Description: A document to notify the deposit of a guarantee.

Processes involved	Tendering
Submitter role	Tenderer
Receiver role	Contracting Authority
Normative schema	xsd/maindoc/UBL-GuaranteeCertificate-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-GuaranteeCertificate-2.1.xsd
RELAX NG schema	rnc/versions/UBL-GuaranteeCertificate-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-GuaranteeCertificate-2.1.ods
Document model (Excel)	mod/maindoc/UBL-GuaranteeCertificate-2.1.xls
Summary report	mod/summary/reports/UBL-GuaranteeCertificate-2.1.html

3.1.29 Instruction for Returns

Description: This document is used to initiate a return of goods. The producer is requesting products which are badly sold either for use in other places or just to free the area from it.

Processes involved	Cyclic Replenishment Program
Submitter role	Seller
Receiver role	Buyer
Normative schema	xsd/maindoc/UBL-InstructionForReturns-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-InstructionForReturns-2.1.xsd
RELAX NG schema	rnc/versions/UBL-InstructionForReturns-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-InstructionForReturns-2.1.ods
Document model (Excel)	mod/maindoc/UBL-InstructionForReturns-2.1.xls
Summary report	mod/summary/reports/UBL-InstructionForReturns-2.1.html
UBL 2.1 example instance	xml/UBL-InstructionForReturns-2.1-Example.xml

3.1.30 Inventory Report

Description: Report about the quantities of each item which are or will be on stock.

Processes involved	Cyclic Replenishment Program
Submitter role	Buyer
Receiver role	Seller
Normative schema	xsd/maindoc/UBL-InventoryReport-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-InventoryReport-2.1.xsd
RELAX NG schema	rnc/versions/UBL-InventoryReport-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-InventoryReport-2.1.ods
Document model (Excel)	mod/maindoc/UBL-InventoryReport-2.1.xls
Summary report	mod/summary/reports/UBL-InventoryReport-2.1.html
UBL 2.1 example instance	xml/UBL-InventoryReport-2.1-Example.xml

3.1.31 Invoice

Description: The document used to request payment.

Processes involved	Billing
Submitter role	Supplier Accounting Party
Receiver role	Customer Accounting Party
Normative schema	xsd/maindoc/UBL-Invoice-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-Invoice-2.1.xsd
RELAX NG schema	rnc/versions/UBL-Invoice-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-Invoice-2.1.ods
Document model (Excel)	mod/maindoc/UBL-Invoice-2.1.xls
Summary report	mod/summary/reports/UBL-Invoice-2.1.html
UBL 2.0 example instance	xml/UBL-Invoice-2.0-Example.xml
UBL 2.1 example instance	xml/UBL-Invoice-2.1-Example.xml

3.1.32 Item Information Request

Description: The document used to request product activity, forecast, or performance data.

Processes involved	Collaborative Planning, Forecasting, and Replenishment
Submitter role	Buyer, Seller
Receiver role	Buyer, Seller
Normative schema	xsd/maindoc/UBL-ItemInformationRequest-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-ItemInformationRequest-2.1.xsd
RELAX NG schema	rnc/versions/UBL-ItemInformationRequest-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-ItemInformationRequest-2.1.ods
Document model (Excel)	mod/maindoc/UBL-ItemInformationRequest-2.1.xls
Summary report	mod/summary/reports/UBL-ItemInformationRequest-2.1.html

3.1.33 Order

Description: The document used to order goods and services.

Processes involved	Ordering
Submitter role	Buyer
Receiver role	Seller
Normative schema	xsd/maindoc/UBL-Order-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-Order-2.1.xsd
RELAX NG schema	rnc/versions/UBL-Order-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-Order-2.1.ods
Document model (Excel)	mod/maindoc/UBL-Order-2.1.xls
Summary report	mod/summary/reports/UBL-Order-2.1.html
UBL 2.0 example instance	xml/UBL-Order-2.0-Example.xml
UBL 2.1 example instance	xml/UBL-Order-2.1-Example.xml

3.1.34 Order Cancellation

Description: The document used to cancel an entire Order.

Processes involved	Ordering, Fulfilment
Submitter role	Buyer
Receiver role	Seller
Normative schema	xsd/maindoc/UBL-OrderCancellation-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-OrderCancellation-2.1.xsd
RELAX NG schema	rnc/versions/UBL-OrderCancellation-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-OrderCancellation-2.1.ods
Document model (Excel)	mod/maindoc/UBL-OrderCancellation-2.1.xls
Summary report	mod/summary/reports/UBL-OrderCancellation-2.1.html
UBL 2.1 example instance	xml/UBL-OrderCancellation-2.1-Example.xml

3.1.35 Order Change

Description: The document used to specify changes to an existing Order.

Processes involved	Ordering, Fulfilment
Submitter role	Buyer
Receiver role	Seller
Normative schema	xsd/maindoc/UBL-OrderChange-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-OrderChange-2.1.xsd
RELAX NG schema	rnc/versions/UBL-OrderChange-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-OrderChange-2.1.ods
Document model (Excel)	mod/maindoc/UBL-OrderChange-2.1.xls
Summary report	mod/summary/reports/UBL-OrderChange-2.1.html
UBL 2.1 example instance	xml/UBL-OrderChange-2.1-Example.xml

3.1.36 Order Response

Description: The document used to indicate detailed acceptance or rejection of an Order or to make a counter-offer.

Processes involved	Ordering
Submitter role	Seller
Receiver role	Buyer
Normative schema	xsd/maindoc/UBL-OrderResponse-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-OrderResponse-2.1.xsd
RELAX NG schema	rnc/versions/UBL-OrderResponse-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-OrderResponse-2.1.ods
Document model (Excel)	mod/maindoc/UBL-OrderResponse-2.1.xls
Summary report	mod/summary/reports/UBL-OrderResponse-2.1.html
UBL 2.1 example instance	xml/UBL-OrderResponse-2.1-Example.xml

3.1.37 Order Response Simple

Description: The document used to indicate simple acceptance or rejection of an entire Order.

Processes involved	Ordering
Submitter role	Seller
Receiver role	Buyer
Normative schema	xsd/maindoc/UBL-OrderResponseSimple-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-OrderResponseSimple-2.1.xsd
RELAX NG schema	rnc/versions/UBL-OrderResponseSimple-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-OrderResponseSimple-2.1.ods
Document model (Excel)	mod/maindoc/UBL-OrderResponseSimple-2.1.xls
Summary report	mod/summary/reports/UBL-OrderResponseSimple-2.1.html
UBL 2.0 example instance	xml/UBL-OrderResponseSimple-2.0-Example.xml
UBL 2.1 example instance	xml/UBL-OrderResponseSimple-2.1-Example.xml

3.1.38 Packing List

Description: A document stating the detail of how goods are packed.

Processes involved	Freight Management
Submitter role	Consignor
Receiver role	Freight Forwarder
Normative schema	xsd/maindoc/UBL-PackingList-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-PackingList-2.1.xsd
RELAX NG schema	rnc/versions/UBL-PackingList-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-PackingList-2.1.ods
Document model (Excel)	mod/maindoc/UBL-PackingList-2.1.xls
Summary report	mod/summary/reports/UBL-PackingList-2.1.html

3.1.39 Performance History

Description: Performance History represents a collection of values gathered for key performance metrics in the trading partner relationship.

Processes involved	Cyclic Replenishment Program
Submitter role	Buyer, Seller
Receiver role	Buyer, Seller
Normative schema	xsd/maindoc/UBL-PerformanceHistory-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-PerformanceHistory-2.1.xsd
RELAX NG schema	rnc/versions/UBL-PerformanceHistory-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-PerformanceHistory-2.1.ods
Document model (Excel)	mod/maindoc/UBL-PerformanceHistory-2.1.xls
Summary report	mod/summary/reports/UBL-PerformanceHistory-2.1.html
UBL 2.1 example instance	xml/UBL-PerformanceHistory-2.1-Example.xml

3.1.40 Prior Information Notice

Description: The document used for a Contracting Party to declare the intention to buy goods, services or works during an specified period.

Processes involved	Tendering
Submitter role	Contracting Authority
Receiver role	Tenderer
Normative schema	xsd/maindoc/UBL-PriorInformationNotice-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-PriorInformationNotice-2.1.xsd
RELAX NG schema	rnc/versions/UBL-PriorInformationNotice-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-PriorInformationNotice-2.1.ods
Document model (Excel)	mod/maindoc/UBL-PriorInformationNotice-2.1.xls
Summary report	mod/summary/reports/UBL-PriorInformationNotice-2.1.html

3.1.41 Product Activity

Description: Represents the movement of a product through a location in terms of the base unit of measure for the item.

Processes involved	CPFR, VMI
Submitter role	Buyer
Receiver role	Seller
Normative schema	xsd/maindoc/UBL-ProductActivity-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-ProductActivity-2.1.xsd
RELAX NG schema	rnc/versions/UBL-ProductActivity-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-ProductActivity-2.1.ods
Document model (Excel)	mod/maindoc/UBL-ProductActivity-2.1.xls
Summary report	mod/summary/reports/UBL-ProductActivity-2.1.html
UBL 2.1 example instance 1	xml/UBL-ProductActivity-2.1-Example-1.xml
UBL 2.1 example instance 2	xml/UBL-ProductActivity-2.1-Example-2.xml
UBL 2.1 example instance 3	xml/UBL-ProductActivity-2.1-Example-3.xml

3.1.42 Quotation

Description: The document used to quote for the provision of goods and services.

Processes involved	Quotation
Submitter role	Seller
Receiver role	Originator
Normative schema	xsd/maindoc/UBL-Quotation-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-Quotation-2.1.xsd
RELAX NG schema	rnc/versions/UBL-Quotation-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-Quotation-2.1.ods
Document model (Excel)	mod/maindoc/UBL-Quotation-2.1.xls
Summary report	mod/summary/reports/UBL-Quotation-2.1.html

UBL 2.0 example instance	xml/UBL-Quotation-2.0-Example.xml
UBL 2.1 example instance	xml/UBL-Quotation-2.1-Example.xml

3.1.43 Receipt Advice

Description: The document used to describe the receipt of goods and services.

Processes involved	Fulfilment
Submitter role	Delivery
Receiver role	Despatch
Normative schema	xsd/maindoc/UBL-ReceiptAdvice-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-ReceiptAdvice-2.1.xsd
RELAX NG schema	rnc/versions/UBL-ReceiptAdvice-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-ReceiptAdvice-2.1.ods
Document model (Excel)	mod/maindoc/UBL-ReceiptAdvice-2.1.xls
Summary report	mod/summary/reports/UBL-ReceiptAdvice-2.1.html
UBL 2.0 example instance	xml/UBL-ReceiptAdvice-2.0-Example.xml

3.1.44 Reminder

Description: The document used to remind the customer of payments overdue.

Processes involved	Billing
Submitter role	Supplier Accounting Party and/or Payee
Receiver role	Customer Accounting Party and/or Payee
Normative schema	xsd/maindoc/UBL-Reminder-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-Reminder-2.1.xsd
RELAX NG schema	rnc/versions/UBL-Reminder-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-Reminder-2.1.ods
Document model (Excel)	mod/maindoc/UBL-Reminder-2.1.xls
Summary report	mod/summary/reports/UBL-Reminder-2.1.html
UBL 2.1 example instance	xml/UBL-Reminder-2.1-Example.xml

3.1.45 Remittance Advice

Description: The document used to specify details of an actual payment.

Processes involved	Payment
Submitter role	Supplier Accounting Party and/or Payee
Receiver role	Customer Accounting Party and/or Payee
Normative schema	xsd/maindoc/UBL-RemittanceAdvice-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-RemittanceAdvice-2.1.xsd
RELAX NG schema	rnc/versions/UBL-RemittanceAdvice-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-RemittanceAdvice-2.1.ods
Document model (Excel)	mod/maindoc/UBL-RemittanceAdvice-2.1.xls
Summary report	mod/summary/reports/UBL-RemittanceAdvice-2.1.html

UBL 2.0 example instance	xml/UBL-RemittanceAdvice-2.0-Example.xml

3.1.46 Request for Quotation

Description: The document used to request a Quotation for goods and services from a Seller.

Processes involved	Quotation
Submitter role	Originator
Receiver role	Seller
Normative schema	xsd/maindoc/UBL-RequestForQuotation-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-RequestForQuotation-2.1.xsd
RELAX NG schema	rnc/versions/UBL-RequestForQuotation-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-RequestForQuotation-2.1.ods
Document model (Excel)	mod/maindoc/UBL-RequestForQuotation-2.1.xls
Summary report	mod/summary/reports/UBL-RequestForQuotation-2.1.html
UBL 2.0 example instance	xml/UBL-RequestForQuotation-2.0-Example.xml
UBL 2.1 example instance	xml/UBL-RequestForQuotation-2.1-Example.xml

3.1.47 Retail Event

Description: The document used to specify basic information about a retail event, including version number, creation date, and time.

Processes involved	Cyclic Replenishment Program
Submitter role	Buyer, Seller
Receiver role	Buyer, Seller
Normative schema	xsd/maindoc/UBL-RetailEvent-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-RetailEvent-2.1.xsd
RELAX NG schema	rnc/versions/UBL-RetailEvent-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-RetailEvent-2.1.ods
Document model (Excel)	mod/maindoc/UBL-RetailEvent-2.1.xls
Summary report	mod/summary/reports/UBL-RetailEvent-2.1.html
UBL 2.1 example instance	xml/UBL-RetailEvent-2.1-Example.xml

3.1.48 Self Billed Credit Note

Description: The Credit Note created by the Debtor in a Self Billing arrangement with a Creditor; Self Billed Credit Note replaces Debit Note in such arrangements.

Processes involved	Billing
Submitter role	Customer Accounting Party
Receiver role	Supplier Accounting Party
Normative schema	xsd/maindoc/UBL-SelfBilledCreditNote-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-SelfBilledCreditNote-2.1.xsd
RELAX NG schema	rnc/versions/UBL-SelfBilledCreditNote-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-SelfBilledCreditNote-2.1.ods

Document model (Excel)	mod/maindoc/UBL-SelfBilledCreditNote-2.1.xls
Summary report	mod/summary/reports/UBL-SelfBilledCreditNote-2.1.html
UBL 2.1 example instance	xml/UBL-SelfBilledCreditNote-2.1-Example.xml

3.1.49 Self Billed Invoice

Description: The Invoice document created by the Customer (rather than the Supplier) in a Self Billing relationship.

Processes involved	Billing
Submitter role	Customer Accounting Party
Receiver role	Supplier Accounting Party
Normative schema	xsd/maindoc/UBL-SelfBilledInvoice-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-SelfBilledInvoice-2.1.xsd
RELAX NG schema	rnc/versions/UBL-SelfBilledInvoice-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-SelfBilledInvoice-2.1.ods
Document model (Excel)	mod/maindoc/UBL-SelfBilledInvoice-2.1.xls
Summary report	mod/summary/reports/UBL-SelfBilledInvoice-2.1.html

3.1.50 Statement

Description: The document used to specify the status of Orders, Billing, and Payment. This document is a Statement of Account and not intended as a summary Invoice.

Processes involved	Billing
Submitter role	Supplier Accounting Party
Receiver role	Customer Accounting Party
Normative schema	xsd/maindoc/UBL-Statement-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-Statement-2.1.xsd
RELAX NG schema	rnc/versions/UBL-Statement-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-Statement-2.1.ods
Document model (Excel)	mod/maindoc/UBL-Statement-2.1.xls
Summary report	mod/summary/reports/UBL-Statement-2.1.html
UBL 2.0 example instance	xml/UBL-Statement-2.0-Example.xml

3.1.51 Stock Availability Report

Description: Report about the quantities of each item which are or will be on stock.

Processes involved	Cyclic Replenishment Program
Submitter role	Seller (Producer)
Receiver role	Buyer (Retailer)
Normative schema	xsd/maindoc/UBL-StockAvailabilityReport-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-StockAvailabilityReport-2.1.xsd
RELAX NG schema	rnc/versions/UBL-StockAvailabilityReport-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-StockAvailabilityReport-2.1.ods

Document model (Excel)	mod/maindoc/UBL-StockAvailabilityReport-2.1.xls
Summary report	mod/summary/reports/UBL-StockAvailabilityReport-2.1.html
UBL 2.1 example instance	xml/UBL-StockAvailabilityReport-2.1-Example.xml

3.1.52 Tender

Description: A message which a tenderer offers a tender to the tendering organization for bid.

Processes involved	Tendering
Submitter role	Tenderer
Receiver role	Contracting Authority
Normative schema	xsd/maindoc/UBL-Tender-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-Tender-2.1.xsd
RELAX NG schema	rnc/versions/UBL-Tender-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-Tender-2.1.ods
Document model (Excel)	mod/maindoc/UBL-Tender-2.1.xls
Summary report	mod/summary/reports/UBL-Tender-2.1.html

3.1.53 Tenderer Qualification

Description: A document used for the Tenderer to declare things about his own condition.

Processes involved	Tendering
Submitter role	Tenderer
Receiver role	Contracting Authority
Normative schema	xsd/maindoc/UBL-TendererQualification-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-TendererQualification-2.1.xsd
RELAX NG schema	rnc/versions/UBL-TendererQualification-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-TendererQualification-2.1.ods
Document model (Excel)	mod/maindoc/UBL-TendererQualification-2.1.xls
Summary report	mod/summary/reports/UBL-TendererQualification-2.1.html

3.1.54 Tenderer Qualification Response

Description: A message which the procurement organization sends to an economic operator in order to notify its admision or exclusion to/from the tendering process.

Processes involved	Tendering
Submitter role	Contracting Authority
Receiver role	Tenderer
Normative schema	xsd/maindoc/UBL-TendererQualificationResponse-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-TendererQualificationResponse-2.1.xsd
RELAX NG schema	rnc/versions/UBL-TendererQualificationResponse-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-TendererQualificationResponse-2.1.ods
Document model (Excel)	mod/maindoc/UBL-TendererQualificationResponse-2.1.xls
Summary report	mod/summary/reports/UBL-TendererQualificationResponse-2.1.html

3.1.55 Tender Receipt

Description: A message sent by the Contracting Party to an Economic Operator in order to notify the reception of the tendering offer.

Processes involved	Tendering
Submitter role	Contracting Authority
Receiver role	Tenderer
Normative schema	xsd/maindoc/UBL-TenderReceipt-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-TenderReceipt-2.1.xsd
RELAX NG schema	rnc/versions/UBL-TenderReceipt-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-TenderReceipt-2.1.ods
Document model (Excel)	mod/maindoc/UBL-TenderReceipt-2.1.xls
Summary report	mod/summary/reports/UBL-TenderReceipt-2.1.html

3.1.56 Trade Item Location Profile

Description: This document is used to send trade item attributes which are focused on replenishment policies.

Processes involved	Collaborative Planning, Forecasting and Replenishment
Submitter role	Buyer, Seller
Receiver role	Buyer, Seller
Normative schema	xsd/maindoc/UBL-TradeItemLocationProfile-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-TradeItemLocationProfile-2.1.xsd
RELAX NG schema	rnc/versions/UBL-TradeItemLocationProfile-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-TradeItemLocationProfile-2.1.ods
Document model (Excel)	mod/maindoc/UBL-TradeItemLocationProfile-2.1.xls
Summary report	mod/summary/reports/UBL-TradeItemLocationProfile-2.1.html
UBL 2.1 example instance	xml/UBL-TradeItemLocationProfile-2.1-Example.xml

3.1.57 Transport Execution Plan

Description: A document which is used by a Transport User and a Transport Service Provider to confirm and describe the details of an agreed upon transport service.

Processes involved	Intermodal Freight Management
Submitter role	Transport Service Provider, Transport User
Receiver role	Transport User, Transport Service Provider
Normative schema	xsd/maindoc/UBL-TransportExecutionPlan-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-TransportExecutionPlan-2.1.xsd
RELAX NG schema	rnc/versions/UBL-TransportExecutionPlan-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-TransportExecutionPlan-2.1.ods
Document model (Excel)	mod/maindoc/UBL-TransportExecutionPlan-2.1.xls
Summary report	mod/summary/reports/UBL-TransportExecutionPlan-2.1.html
UBL 2.1 example instance	xml/UBL-TransportExecutionPlan-2.1-Example.xml

3.1.58 Transport Execution Plan Request

Description: A document which initiates the negotiation of a transport service between a Transport User and a Transport Service Provider.

Processes involved	Intermodal Freight Management
Submitter role	Transport User
Receiver role	Transport Service Provider
Normative schema	xsd/maindoc/UBL-TransportExecutionPlanRequest-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-TransportExecutionPlanRequest-2.1.xsd
RELAX NG schema	rnc/versions/UBL-TransportExecutionPlanRequest-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-TransportExecutionPlanRequest-2.1.ods
Document model (Excel)	mod/maindoc/UBL-TransportExecutionPlanRequest-2.1.xls
Summary report	mod/summary/reports/UBL-TransportExecutionPlanRequest-2.1.html
UBL 2.1 example instance	xml/UBL-TransportExecutionPlanRequest-2.1-Example.xml

3.1.59 Transport Progress Status

Description: A document sent from a Transportation Network Manager to a Transport Service Provider giving the status of a transport means.

Processes involved	Intermodal Freight Management
Submitter role	Transportation Network Manager
Receiver role	Transport Service Provider
Normative schema	xsd/maindoc/UBL-TransportProgressStatus-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-TransportProgressStatus-2.1.xsd
RELAX NG schema	rnc/versions/UBL-TransportProgressStatus-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-TransportProgressStatus-2.1.ods
Document model (Excel)	mod/maindoc/UBL-TransportProgressStatus-2.1.xls
Summary report	mod/summary/reports/UBL-TransportProgressStatus-2.1.html
UBL 2.1 example instance	xml/UBL-TransportProgressStatus-2.1-Example.xml

3.1.60 Transport Progress Status Request

Description: A document sent from a Transport Service Provider to a Transport Network Manager requesting the status of a particular transport means.

Processes involved	Intermodal Freight Management
Submitter role	Transport Service Provider
Receiver role	Transportation Network Manager
Normative schema	xsd/maindoc/UBL-TransportProgressStatusRequest-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-TransportProgressStatusRequest-2.1.xsd
RELAX NG schema	rnc/versions/UBL-TransportProgressStatusRequest-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-TransportProgressStatusRequest-2.1.ods
Document model (Excel)	mod/maindoc/UBL-TransportProgressStatusRequest-2.1.xls
Summary report	mod/summary/reports/UBL-TransportProgressStatusRequest-2.1.html

UBL 2.1 example instance	xml/UBL-TransportProgressStatusRequest-2.1-Example.xml

3.1.61 Transport Service Description

Description: A document sent from the Transport Service Provider to the Transport User in order to announce a transport service.

Processes involved	Intermodal Freight Management
Submitter role	Transport Service Provider
Receiver role	Transport User
Normative schema	xsd/maindoc/UBL-TransportServiceDescription-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-TransportServiceDescription-2.1.xsd
RELAX NG schema	rnc/versions/UBL-TransportServiceDescription-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-TransportServiceDescription-2.1.ods
Document model (Excel)	mod/maindoc/UBL-TransportServiceDescription-2.1.xls
Summary report	mod/summary/reports/UBL-TransportServiceDescription-2.1.html
UBL 2.1 example instance	xml/UBL-TransportServiceDescription-2.1-Example.xml

3.1.62 Transport Service Description Request

Description: A document sent from a Transport User to a Transport Service Provider to request relevant transport services announced by the Transport Service Provider.

Processes involved	Intermodal Freight Management
Submitter role	Transport User
Receiver role	Transport Service Provider
Normative schema	xsd/maindoc/UBL-TransportServiceDescriptionRequest-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-TransportServiceDescriptionRequest-2.1.xsd
RELAX NG schema	rnc/versions/UBL-TransportServiceDescriptionRequest-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-TransportServiceDescriptionRequest-2.1.ods
Document model (Excel)	mod/maindoc/UBL-TransportServiceDescriptionRequest-2.1.xls
Summary report	mod/summary/reports/UBL-TransportServiceDescriptionRequest-2.1.html
UBL 2.1 example instance	xml/UBL-TransportServiceDescriptionRequest-2.1-Example.xml

3.1.63 Transportation Status

Description: A message to report the transport status and/or change in the transport status (i.e., an event) between agreed parties.

Processes involved	Freight Status Reporting
Submitter role	Transport Service Provider
Receiver role	Transport User
Normative schema	xsd/maindoc/UBL-TransportationStatus-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-TransportationStatus-2.1.xsd
RELAX NG schema	rnc/versions/UBL-TransportationStatus-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-TransportationStatus-2.1.ods

Document model (Excel)	mod/maindoc/UBL-TransportationStatus-2.1.xls
Summary report	mod/summary/reports/UBL-TransportationStatus-2.1.html
UBL 2.1 example instance	xml/UBL-TransportationStatus-2.1-Example.xml

3.1.64 Transportation Status Request

Description: A message to request transportation status and/or change in the transportation status (i.e., an event) between agreed parties.

Processes involved	Freight Status Reporting
Submitter role	Transport User
Receiver role	Transport Service Provider
Normative schema	xsd/maindoc/UBL-TransportationStatusRequest-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-TransportationStatusRequest-2.1.xsd
RELAX NG schema	rnc/versions/UBL-TransportationStatusRequest-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-TransportationStatusRequest-2.1.ods
Document model (Excel)	mod/maindoc/UBL-TransportationStatusRequest-2.1.xls
Summary report	mod/summary/reports/UBL-TransportationStatusRequest-2.1.html
UBL 2.1 example instance	xml/UBL-TransportationStatusRequest-2.1-Example.xml

3.1.65 Unawarded Notification

Description: The document used to communicate that a contract has been awarded to another tenderer.

Processes involved	Tendering
Submitter role	Contracting Authority
Receiver role	Tenderer
Normative schema	xsd/maindoc/UBL-UnawardedNotification-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-UnawardedNotification-2.1.xsd
RELAX NG schema	rnc/versions/UBL-UnawardedNotification-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-UnawardedNotification-2.1.ods
Document model (Excel)	mod/maindoc/UBL-UnawardedNotification-2.1.xls
Summary report	mod/summary/reports/UBL-UnawardedNotification-2.1.html

3.1.66 Utility Statement

Description: The Utility Statement contains information on the consumption of services provided by utility suppliers to private and public customers. These utilities include electricity, gas, water and telephony services. The Utility Statement is therefore a supplement to an Invoice or Credit Note.

Processes involved	Utility Billing
Submitter role	Supplier Accounting Party
Receiver role	Customer Accounting Party
Normative schema	xsd/maindoc/UBL-UtilityStatement-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-UtilityStatement-2.1.xsd
RELAX NG schema	rnc/versions/UBL-UtilityStatement-2.1.rnc

Document model (ODF)	mod/maindoc/UBL-UtilityStatement-2.1.ods
Document model (Excel)	mod/maindoc/UBL-UtilityStatement-2.1.xls
Summary report	mod/summary/reports/UBL-UtilityStatement-2.1.html

3.1.67 Waybill

Description: The Waybill document states the details of the transportation, charges, and terms and conditions under which a transportation service is provided. See Section 2.15.4, "Waybill" and compare with Section 2.15.3, "Bill of Lading".

Processes involved	Freight Management
Submitter role	Freight Forwarder, Carrier
Receiver role	Consignor (or Consignee), Freight Forwarder
Normative schema	xsd/maindoc/UBL-Waybill-2.1.xsd
Runtime schema	xsdrt/maindoc/UBL-Waybill-2.1.xsd
RELAX NG schema	rnc/versions/UBL-Waybill-2.1.rnc
Document model (ODF)	mod/maindoc/UBL-Waybill-2.1.ods
Document model (Excel)	mod/maindoc/UBL-Waybill-2.1.xls
Summary report	mod/summary/reports/UBL-Waybill-2.1.html
UBL 2.0 example instance	xml/UBL-Waybill-2.0-Example-International.xml

3.2 UBL 2.1 Common Schemas

The xsd/common directory contains schemas referenced by the document schemas in xsd/maindoc. Elements defined in the common schemas constitute a library of reusable business data components from which the UBL document schemas are assembled.

The name of each schema file together with a brief description of its contents is given below.

3.2.1 Reusable BIE Schemas

CommonBasicComponents
xsd/common/UBL-CommonBasicComponents-2.1.xsd

The CommonBasicComponents schema defines the global Basic Business Information Entities (BBIEs) that are used throughout UBL, serving, in effect, as a "global BBIE type database" for constructing documents. BBIEs are the "leaf nodes" of UBL documents, corresponding to individual data fields in traditional printed business forms.

CommonAggregateComponents
xsd/common/UBL-CommonAggregateComponents-2.1.xsd

The CommonAggregateComponents schema defines the Aggregate Business Information Entities (ABIEs) that are used throughout UBL, serving, in effect, as an "ABIE type database" for constructing the main documents.

3.2.2 Reusable Data Type Schemas

CCTS_CCT_SchemaModule
xsd/common/CCTS_CCT_SchemaModule-2.1.xsd

This schema provides Core Component Types as defined by [**CCTS**]. These types are used to construct higher-level data types in a standardized and consistent manner. This schema is defined by UN/CEFACT and should not be modified. It is imported by the UBL Unqualified Data Type Schema, and its types are the basis upon which UBL's unqualified data types are defined.

UnqualifiedDataTypes
`xsd/common/UBL-UnqualifiedDataTypes-2.1.xsd`

This schema defines Unqualified Data Types for BBIE definition. These types are derived from the Core Component Types in `CCTS_CCT_SchemaModule`. Where an unqualified type is not based solely on an XSD data type, all CCTS supplementary components are made available in the UBL UDT from the CCTS CCT.

QualifiedDataTypes
`xsd/common/UBL-QualifiedDataTypes-2.1.xsd`

[**CCTS**] permits the definition of Qualified Datatypes as derivations from CCTS-specified Unqualified Datatypes. In UBL 2.1, all data type qualifications are expressed in the [**CVA**] file cva/UBL-DefaultDTQ-2.1.cva. The UBL-QualifiedDataTypes-2.1.xsd file in the UBL 2.1 release is included among the schema modules imported by the Common Library and all document-level schema fragments in order to be consistent with the relationship of types in a CCTS framework, though the schema module itself has no declarations.

See Appendix E, *Data Type Qualifications in UBL (Non-Normative)* for information regarding UBL 2.1 data type derivation.

3.2.3 Documentation Metadata Schema

CoreComponentParameters
`xsd/common/UBL-CoreComponentParameters-2.1.xsd`

The CoreComponentParameters schema defines the structure of the annotation/documentation sections that appear in all the other schemas, providing a consistent format for metadata such as object class, representation terms, semantic descriptions, and other supplementary information.

While not required by UBL schemas, this module is provided to encourage consistency in the documentation of customized extensions.

3.2.4 Extension Content Schemas

UBL extensions enable the validation of user-defined additions to the standard schemas, which are sometimes needed to satisfy legal requirements and can perform other useful functions as well. UBL 2.1 schemas are supplied with a predefined standard extension that supports advanced digital signatures; see Section 5.3, "UBL Extension for Enveloped XML Digital Signatures" and Section 3.3, "Schema Dependencies". For further information regarding the UBL extension mechanism, see [**Customization**].

CommonExtensionComponents
`xsd/common/UBL-CommonExtensionComponents-2.1.xsd`

The CommonExtensionComponents schema defines the extension structures that are used in all UBL document types, providing metadata regarding the use of an extension embedded in a UBL document instance.

ExtensionContentDatatype
`xsd/common/UBL-ExtensionContentDataType-2.1.xsd`

The ExtensionContentDataType schema specifies the actual structural constraints of the extension element containing the foreign non-UBL content. This is delivered as both a functional component and illustration of the definition of an extension schema by importing the UBL Signature Extension module and namespace. To support the constraints of additional extension structures, this content module is augmented with other schema import directives. No changes are required to the complex type declaration. Note that the constraints of only the imported constructs are being declared, not the constraints of unknown constructs. Constructs for which there is no declaration are not constrained by validation.

3.2.5 Signature Extension Schemas

See Section 5.3, "UBL Extension for Enveloped XML Digital Signatures" for further information regarding the UBL extension supporting digital signatures such as XAdES.

CommonSignatureComponents
`xsd/common/UBL-CommonSignatureComponents-2.1.xsd`

The CommonSignatureComponents schema defines the scaffolding structures containing the IETF/W3C Digital Signature information XML elements related to either the entire document or particular signature business objects found within the document.

XAdESv132
`xsd/common/UBL-XAdESv132-2.1.xsd`

This is a copy of the XAdES v1.3.2 schema file [http://uri.etsi.org/01903/v1.3.2/XAdES.xsd], modified only to change the importing URI for the XML digital signature core schema file.

The presence of this schema file does not oblige the use of XAdES. It is provided only as a convenience for those users who choose to include an XAdES extension inside of a digital signature.

XAdESv141
`xsd/common/UBL-XAdESv141-2.1.xsd`

This is a copy of the XAdES v1.4.1 schema file [http://uri.etsi.org/01903/v1.4.1/XAdESv141.xsd], modified only to change the importing URI for the XAdES v1.3.2 schema file.

The presence of this schema file does not oblige the use of XAdES. It is provided only as a convenience for those users who choose to include an XAdES extension inside of a digital signature.

xmldsig-core-schema
`xsd/common/UBL-xmldsig-core-schema-2.1.xsd`

This is a copy of the IETF/W3C Digital Signature core schema file [http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd], modified only to remove the unnecessary PUBLIC and SYSTEM identifiers from the DOCTYPE.

3.2.6 Signature Components

SignatureBasicComponents
`xsd/common/UBL-SignatureBasicComponents-2.1.xsd`

The SignatureBasicComponents schema defines those Basic Business Information Entities (BBIEs) that are used for signature constructs not defined in the common

library. BBIEs are the “leaf nodes” of UBL documents, expressing simple string values.

SignatureAggregateComponents

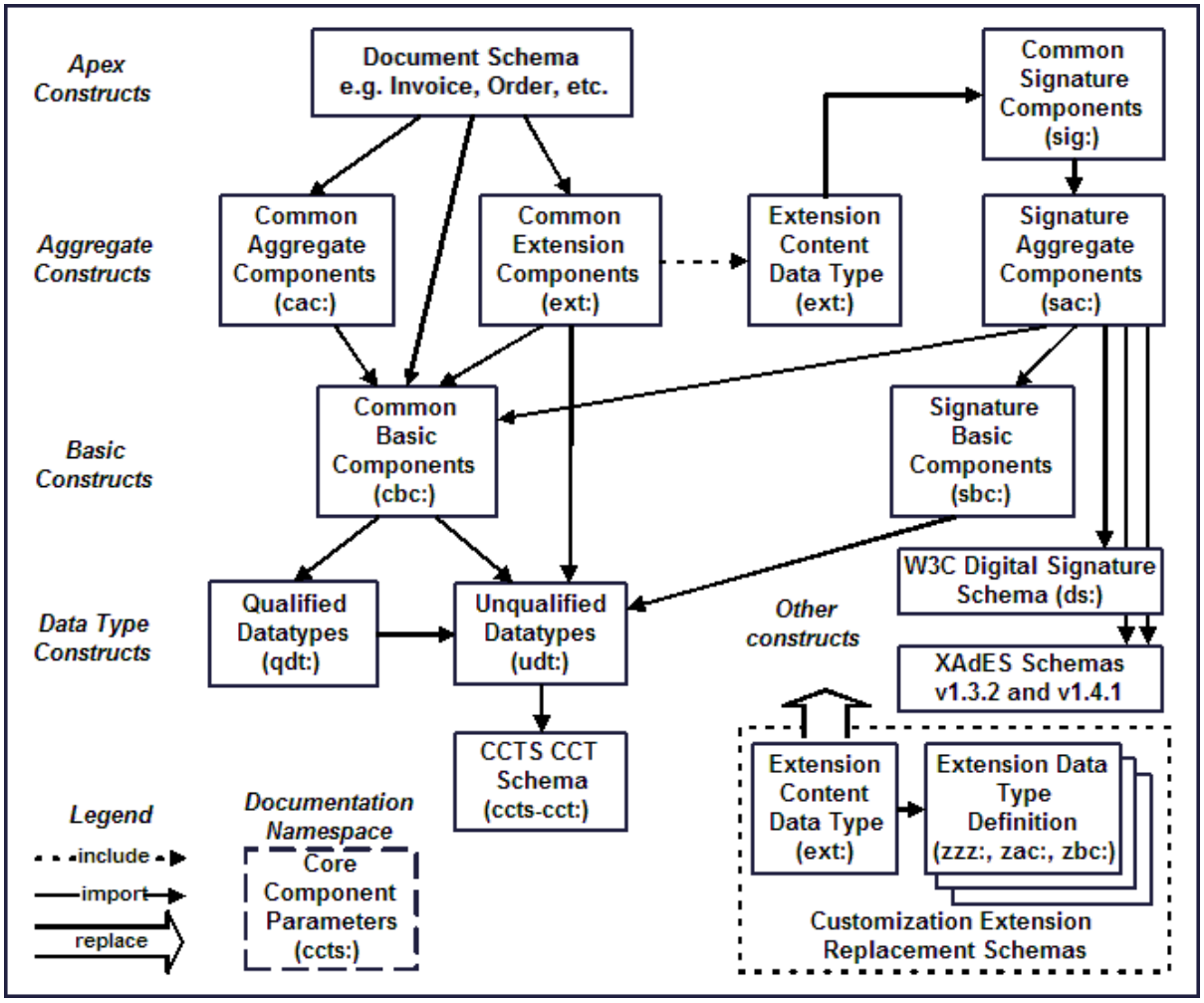
`xsd/common/UBL-SignatureAggregateComponents-2.1.xsd`

The SignatureAggregateComponents schema defines those Aggregate Business Information Entities (ABIEs) that are used for signature constructs not defined in the common library. ABIEs are the “branch nodes” of UBL documents, expressing component values composed of leaf nodes and other branch nodes.

3.3 Schema Dependencies

The following diagram shows the dependencies among the schema modules making up a UBL 2.1 document schema.

Figure 63. UBL Schema Dependencies



The UBL schemas are delivered supporting the UBL standardized extension for digital signatures, defining the content of the extension to be a single element either in or out of the UBL signature extension namespace. As shown on the bottom right in this diagram, a set of UBL schemas supporting a different user-customized extension is created by replacing the delivered ExtensionContentDataType schema fragment with one also importing the required custom schema fragments that define the custom content. For more regarding the signature extension, see Section 5.3, "UBL Extension for Enveloped XML Digital Signatures".

The relationship of the UBL schemas to the UBL data model is illustrated in Figure C.1, "UBL Spreadsheet Realization".

4 Additional Document Constraints

In addition to the UBL 2.1 document constraints formally expressed by the schemas in Section 3, “UBL 2.1 Schemas”, UBL mandates several other rules governing conformant UBL 2.1 instances that cannot be expressed using W3C Schema. These additional UBL document rules, addressing instance validation, character encoding, and empty elements, are specified below.

These rules first appeared in the OASIS UBL 1.0 and UBL 1.0 NDR Standards. They are listed here because logically they belong with the great majority of UBL instance constraints specified in the schemas. To aid in coordinating references between these various publications, the rules below retain their original “IND” labels. The former IND4 was removed in the revision process leading to UBL 2.0.

4.1 Validation

The UBL library and document schemas are targeted at supporting business information exchanges. Business information exchanges require a high degree of precision to ensure that application processing and corresponding business cycle actions are reflective of the purpose, intent, and information content agreed to by both trading partners. Schemas provide the necessary mechanism for ensuring that instance documents do in fact support these requirements.

> **[IND1]** All UBL instance documents MUST validate to a corresponding schema.

4.2 Character Encoding

XML supports a wide variety of character encodings. Processors must understand which character encoding is employed in each XML document. XML 1.0 supports a default value of UTF-8 for character encoding, but best practice is to always identify the character encoding being employed.

> **[IND2]** All UBL instance documents MUST identify their character encoding within the XML declaration.

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
```

UBL, as an OASIS TC, is obligated to conform to agreements OASIS has entered into. OASIS is a liaison member of the ISO IEC ITU UN/CEFACT eBusiness Memorandum of Understanding Management Group (MOUMG). Resolution 01/08 (MOU/MG01n83) requires the use of UTF-8.

> **[IND3]** In conformance with ISO IEC ITU UN/CEFACT eBusiness Memorandum of Understanding Management Group (MOUMG) Resolution 01/08 (MOU/MG01n83) as agreed to by OASIS, all UBL XML SHOULD be expressed using UTF-8.

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
```

4.3 Empty Elements

Use of empty elements within XML instance documents is a source of controversy for a variety of reasons. An empty element does not simply represent data that is missing. It may express data that is not applicable for some reason, trigger the expression of an attribute, denote all possible values instead of just one, mark the end of a series of data, or appear as a result of an error in XML file generation. Conversely, missing data elements can also have meaning—data not provided by a trading partner. In information exchange environments, different trading partners may allow, require, or ban empty elements. UBL has

determined that empty elements do not provide the level of assurance necessary for business information exchanges and therefore will not be used.

> **[IND5]** UBL conformant instance documents MUST NOT contain an element devoid of content or containing null values, except in the case of extension, where the UBL ExtensionContent element is used.

To ensure that no attempt is made to circumvent rule IND5, UBL also prohibits attempting to convey meaning by not conveying an element.

> **[IND6]** The absence of a construct or data in a UBL instance document MUST NOT carry meaning.

5 UBL Digital Signatures

This section provides context for UBL digital signatures and then defines profiles for advanced digital signatures in UBL and a specific UBL extension that implements one kind of advanced digital signature.

5.1 Introduction (Non-Normative)

There are certain circumstances in which it becomes necessary to electronically sign UBL documents. This can be the case when creating orders or invoices. In some countries, digitally signing electronic invoices is required by law.

UBL (without extension) has a data structure for defining signatures and a number of elements for using such signatures in a document. To integrate UBL into the larger standards environment, this section associates the IETF/W3C XML Digital Signature specification [**xmldsig**] (a general framework for digitally signing XML documents) with the signature elements provided by UBL, with specific provisions to use extensions supporting [**XAdES**], XML Advanced Electronic Signatures (ETSI TS 101 903), when the electronic signing of UBL documents addresses special advanced legal and technical requirements. XAdES extends XMLDSig for use with advanced and qualified electronic signatures as specified in European Union Directive [**1999/93/EC**]. Use of XAdES and the concept of Advanced Electronic Signature is not limited to Europe, as it is being adopted by many countries outside the EU, and, at the time of publication of this specification, it is undergoing standardization in ISO TC 154 as ISO/CD 14533-2.

One important benefit of XAdES is that it allows the addition of information and timestamps that extend the validity of a signature beyond the expiration or revocation of the electronic certificates involved in signature verification or the obsolescence of the underlying cryptographic keys and algorithms. By extending XMLDSig with additional embedded syntax and processing, XAdES satisfies the European Commission Directive on a Community Framework for Electronic Signatures as well as other use cases requiring long-term preservation of signed documents. XAdES contains several modules that permit various levels of security, such as content commitment and non-repudiation enforcement with timestamps and long-term signature verification.

The work of standardizing electronic signatures was supported by the European Commission and assigned to the Information and Communication Technologies Standards Board (ICTSB), a round table of most European IT standards bodies and some international standards bodies such as the IETF and W3C.

The two UBL digital signature profiles defined further on represent two approaches to signing UBL documents: enveloped and detached. Each of these approaches uses XMLDSig in a way that may or may not include XAdES features. In other words, the mechanisms implemented here can be used not only to implement XAdES in these two ways but also to implement other signature technologies based on XMLDSig as well.

Using UBL Digital Signature Profiles one can conform to, for example, the UN/CEFACT Signed Digital Evidence Interoperability Recommendation [**UN/CEFACT Rec. 37**]. *[To date, this recommendation has not been published by UN/CEFACT.]*

5.1.1 XML Digital Signatures

5.1.1.1 Overview

Digital signatures, when appropriate rules and functions are used, can support the following properties for a document:

- Integrity: the document has not been modified since it was signed.
- Authenticity: the identity of the party creating the signature that applies to the document is certified.

- Non-repudiation (content commitment): the document signer cannot deny its involvement in creating and/or approving the document (depending on the context and signer role).
- Anteriority: associating a time-stamp to the signature, a proof that the signature (and therefore the signed document) existed before a certain point in time.

[**xmldsig**] defines XML Signature processing rules and syntax to provide integrity and message authentication and/or signer authentication services for data of any type, whether located within the XML that includes the signature or elsewhere. [**RFC3161**] specifies a standard format for time-stamping that can be used with XMLDSig and XAdES.

The [**1999/93/EC**] directive defines the following technology-neutral requirements that an electronic signature must meet to be considered an Advanced Electronic Signature (AdES) and have legal validity:

- it is uniquely linked to the signatory;
- it is capable of identifying the signatory;
- it is created using means that the signatory can maintain under his sole control; and
- it is linked to the data to which it relates in such a manner that any subsequent change of the data is detectable.

The Qualified Signature (QS) is also defined as an AdES based on Qualified Certificates (QC) and Secure Signature Creation Devices for signing operations. In Europe, QS is equivalent to handwritten signature provided it is based on a QC issued by an accredited Certificate Service Provider. These references are provided only for informational use and refer to the framework defined in [**1999/93/EC**].

XAdES extends XMLDSig to support AdES, but its adoption is not limited to an EU context, as similar requirements are in place in other countries. The introduction to [**XAdES**] reads, in part,

> The XML advanced electronic signatures defined in the present document will be built by incorporating to the XML signatures as defined in XMLDSIG one new `ds:Object` XML element containing the additional qualifying information.

That XAdES is completely embedded in XMLDSig ensures that the UBL profiles for XMLDSig are sufficient to support XAdES. These profiles also support other existing or future extensions of XMLDSig that are completely embedded in XMLDSig syntax. These other possible UBL digital signature profiles may or may not use the XAdES extensions to XMLDSig.

It is important to note that XAdES and XMLDSig define digital signature processing rules and syntax but do not cover the implementation of security measures required for an AdES, which are out of scope for this document.

Implementation may depend on local regulations in place and specific provisions set by the authority issuing the certificates supporting the signature. The implementer has to determine the set of requirements that apply to the specific context of use and determine accordingly the suitability of the standards and the specific profiles to be used. XAdES can help in fulfilling legal requirements, but this is not just a matter of correctly applying a technical standard. Users are advised to examine the regulations applicable to their specific context of use.

5.1.1.2 XML Signature Types

An XML signature may be (non-exclusively) described (per XMLDSig and XAdES) as detached, enveloping, or enveloped.

- **Detached.** The signature applies to content that is external to the `<ds:Signature>` element and can be identified via a URI or transform. Consequently, the signature is “detached" from the content it signs. This definition typically applies to separate data objects, but it also includes the case where

the <ds:Signature> and signed data object are sibling elements residing within the same XML document.

- **Enveloping.** The signature applies to content found within a <ds:Object> element of the signature itself. The <ds:Object> (or its content) is identified via a <ds:Reference> (using a URI fragment identifier or transform).

- **Enveloped.** The signature applies to the XML content that contains <ds:Signature> as an element. Implementations of enveloped signature(s) must take care not to include the signature in the calculation of the signature value.

UBL defines two profiles for signing a UBL document: enveloped and detached.

5.1.1.3 XAdES

A compliant implementation of XAdES guarantees wide acceptance in implementing legal regulations, such as European Commission Directive [**1999/93/EC**] and European Commission Decision [**2011/130/UE**], and supports best practices in eInvoicing, eProcurement, and eBusiness in general as set forth by relevant standard bodies such as CEN ([**CWA15580**] and [**CWA15579**]).

The UBL implementation of XAdES provides the following additional properties:

- A signed UBL document will be processed correctly by any compliant UBL software (including UBL software that is not XMLDSig/XAdES aware) and by any compliant XMLDSig/XAdES verification software (including software that is not UBL aware)

- No change is required for currently defined UBL or XMLDSig/XAdES syntaxes

- The extension mechanism specified here supports any XMLDSig/XAdES form, leaving to the implementer the choice of the most appropriate one according to the specific legal framework or application context.

XAdES defines a set of forms that extends XMLDSig and allows adding to the signature some validation data.

The two basic forms are:

- **XAdES-BES**, which satisfies the minimum requirements for AdES; and

- **XAdES-EPES**, which builds on XAdES-BES to include a security policy identifier that specifies the rules followed to validate the signature.

A conformant XAdES signature generation and verification implementation supports at least XAdES-BES or XAdES-EPES.

The other forms can be built by the signature generator or the signature verifier by extending one of the two basic forms. They are:

- **XAdES-T**, where a timestamp is added to enforce content commitment (non-repudiation) and as a proof of anteriority. This envelope allows ascertaining the validity of a signature in case the signer certificate is later revoked;

- **XAdES-C**, which adds to the signed document a complete reference to verification data (certificates and revocation lists) to support long-term signature verification;

- **XAdES-X**, which adds timestamps to XAdES-C references to protect against future compromise of certificates;

- **XAdES-X-L**, which is similar to XADES-X but adds real certificates and revocation lists instead of just references; and

- **XAdES-A**, which adds timestamps (periodically, as required) to extend the validity period for long-term storage, taking into account a possible weakening of the algorithms used to sign the document and related certificates during the storage period.

No specific XAdES form is recommended for a UBL document, as this choice depends on the specific context of use, agreements between the parties, and local regulations.

5.1.1.4 Requirements for Digital Signatures in UBL

The main requirements to be addressed when choosing a specific signature profile can be divided into the following categories:

- **Legal requirements.** In some countries a digital signature is required on electronic invoices. It can also be compulsory in electronic procurement, especially in a cross border context, to have digital signature on the key document exchanged, e.g., on orders. Another important legal requirement is long-term document preservation, for a storage period that in general is specific in each country and can span many years. The requirement to guarantee the integrity and authenticity of all fiscally relevant archived documents, as specified, for example, by [**CWA15580**] for electronic invoices, can be met with digital signatures when proper XAdES forms are used.

- **Business requirements.** A digital signature can reduce the risks associated with a business transaction (e.g., content commitment of a commercial order, proof-of-origin and integrity of an invoice), and its use can be provided for in the interchange agreement between parties. The choice of the signature format and its application is a key element for interoperability.

- **Process requirements.** The presence of the digital signature should not add any specific constraints on UBL document content processing. If the signed document remains a valid UBL document, the signature can be verified at any stage of the process: it should be possible to validate a signed document at any time “as is" by UBL and XAdES verifiers.

Archiving of UBL documents also can be an important issue to consider, as document preservation has specific requirements.

5.2 Profiles for UBL Digital Signatures

UBL specifies two profiles for use in digitally signing UBL documents:

- **Enveloped Signature Profile:** One or more signatures are added to the UBL document inside a single identifiable and dedicated UBL Extension. Other UBL extensions MAY be present provided they have different identifiers so that they can be distinguished from the one that contains the document signature(s). This profile is defined such that UBL content processing can be separated from electronic signature processing, both on the issuing side and on the receiving side, and specialized applications can be devoted to each function. The UBL application does not need to be electronic signature aware, and the electronic signature application does not need to be involved in the management of the UBL syntax. A signature business object in the UBL document may reference a particular electronic signature in the extension.

- **Detached Signature Profile:** The signature is outside the UBL document content in another information resource. Some mechanism has to be defined by the implementer to send or make available the signature to the recipient. This method of signing may be identified in the UBL document. This approach can be useful to avoid or minimize any kind of modification to the UBL document and is compatible with other signature methods not explicitly referenced by this profile.

The two profiles for adding one or more digital signatures to a UBL document are based on [**xmldsig**]. These profiles and their associated methods decouple the UBL document to be signed from any specificity in the digital signature standard adopted within XMLDSig. The XAdES standard is an example of a standard use of XMLDSig. UBL users may use any standard built on XMLDSig or simply use XMLDSig as it stands without any extensions.

Managing XML signatures inside of a UBL document is described in Section 5.2.1, "Enveloped XML Signatures in UBL Documents". Managing XML signatures outside of a UBL document is described in Section 5.2.2, "Detached XML Signatures for UBL Documents".

Both profiles support co-signatures, i.e., a UBL document can be independently cosigned by multiple signers in any order and time. Both profiles support countersignatures, i.e., a UBL document can have its signatures signed by another signature. The enveloped signature profile supports a final signature, i.e., a UBL document once signed with a final signature cannot have any other signature added without invalidating the final signature.

The choice of the most suitable profile should take into account mainly the specific document processing and delivery infrastructure.

The main advantage of the enveloped profile is that the signature(s) are embedded in the UBL document (which syntactically remains a valid UBL document). This means that the transport of the signatures is guaranteed by the UBL document delivery infrastructure.

The detached signature profile has a simpler preparation phase and signature procedure, but specific means to send or make available the signature(s) to the recipient have to be implemented. A standard container like [**ODFP**] can be used to associate the UBL document with detached advanced electronic signature(s) that apply to it. The simple [**ASiC**] container (ASiC-S) can be created at a later time than signature generation so that it contains a UBL document and one or more detached signatures that apply to it.

5.2.1 Enveloped XML Signatures in UBL Documents

The enveloped signature profile supports one or more signatures to be applied to a UBL document and embedded in the UBL document itself inside a dedicated extension. This profile can be used with all UBL documents under their respective `<ext:UBLExtensions>` extension points. UBL syntax implementing the enveloped profile, together with examples of its use, are provided in Section 5.3, "UBL Extension for Enveloped XML Digital Signatures".

The user MAY choose to indicate in a `<cac:Signature>` element that the signature details are found in the signature extension. The URI `urn:oasis:names:specification:ubl:dsig:enveloped` is reserved as a value for `<cbc:SignatureMethod>` to signal this. The URI `urn:oasis:names:specification:ubl:dsig:enveloped:xades` MAY be used as a value for `<cbc:SignatureMethod>` to signal when XAdES is in use. Additionally, the user MAY include a `<cbc:ID>` child of `<cac:Signature>` for referencing purposes from the enveloped signature. The identifier used can be any value, but for convenience the URI of a URN beginning with `urn:oasis:names:specification:ubl:signature:` and ending with the local name of the parent of the signature business object and optionally followed with a colon and number, as in the `urn:oasis:names:specification:ubl:signature:IssuerEndorsement` example, is reserved for this purpose for UBL users. As with all identifier references, the referenced identifier SHOULD exist and be unique across all such identifier values. An example is as follows:

```
<cac:Signature>
  <cbc:ID>urn:oasis:names:specification:ubl:signature:Invoice</cbc:ID>
  <cbc:SignatureMethod
    >urn:oasis:names:specification:ubl:dsig:enveloped</cbc:SignatureMethod>
  <cac:SignatoryParty>
    <cac:PartyIdentification>
      <cbc:ID>MyParty</cbc:ID>
    </cac:PartyIdentification>
  </cac:SignatoryParty>
</cac:Signature>
```

See Section 5.4, "Digital Signature Examples" for a sample UBL Invoice that references an enveloped digital signature.

5.2.1.1 Enveloped Signature Syntax and Transformation

There are two distinctive levels of syntax present: UBL-specified scaffolding under the extension point used to contain the signature information and IETF/W3C-specified information for each digital signature.

As well, a transformation element is present to prevent a signature from being invalidated by the subsequent addition of another signature.

These are described in detail in Section 5.3.4, “Structure” and Section 5.3.5, “Transformation”.

5.2.2 Detached XML Signatures for UBL Documents

This profile supports the application to a UBL document of one or more signatures located outside of the document itself in some other resource.

It is important to note that externally signing a UBL document with a detached signature imposes no requirements on the UBL document itself. Such a signature, in any kind of signature container, can digitally sign the content of a UBL document regardless of whether this is reflected in the document.

If a user knows the document will have a detached conformant IETF/W3C XML digital signature, the user MAY choose to signal in their UBL document that it is so signed. The URI value `urn:oasis:names:specification:ubl:dsig:detached` is reserved to indicate that the detached signature is an IETF/W3C XML digital signature. The URI `urn:oasis:names:specification:ubl:dsig:detached:xades` MAY be used as a value to signal when XAdES is in use. The value is used in the `<cbc:SignatureMethod>` child of `<cac:Signature>`.

If the location of the digital signature is known, the user MAY choose to indicate the location in a `<cbc:URI>` child element of a `<cac:ExternalReference>` child element of a `<cac:DigitalSignatureAttachment>` element.

A complete example of a `<cac:Signature>` business object in the UBL instance is:

```
<cac:Signature>
  <cbc:ID>urn:oasis:names:specification:ubl:signature:Invoice</cbc:ID>
  <cbc:SignatureMethod
    >urn:oasis:names:specification:ubl:dsig:detached</cbc:SignatureMethod>
  <cac:SignatoryParty>
    <cac:PartyIdentification>
      <cbc:ID>MyParty</cbc:ID>
    </cac:PartyIdentification>
  </cac:SignatoryParty>
  <cac:DigitalSignatureAttachment>
    <cac:ExternalReference>
      <cbc:URI>sigFile.xml</cbc:URI>
    </cac:ExternalReference>
  </cac:DigitalSignatureAttachment>
</cac:Signature>
```

Note

A document with multiple detached signatures is simply a document that is co-signed. By the appropriate use of the `<ds:Reference>` element pointing to the UBL document from a detached signature file, all such signatures are signing the content of the document but not each other. A *countersigning* document signature, on the other hand, signs signatures already created for and external to or present in the document at the time it is countersigned. A digital countersignature `<ds:Signature>`, which may be located internal to the UBL document or in an external file, includes additional `<ds:Reference>` elements, each pointing either to the `<ds:Signature>` element or `<ds:SignatureValue>` element child of the signature being

signed. In the first case, where the signature is detached, the <ds:Reference> element points to the external file for that signature; in the second case, where the signature is enveloped, the <ds:Reference> element points to the Id= value of either the <ds:Signature> or <ds:SignatureValue> element for that signature.

Note

The XAdES specification supports an alternative countersignature approach where a <ds:Signature> element pointing to the countersigned signature's <ds:SignatureValue> is embedded in the <ds:Object> of the countersigning signature. The inclusion of an alternative method in this specification does not prohibit this approach.

See Section 5.4, “Digital Signature Examples” for a sample UBL Invoice that references a detached digital signature.

5.2.2.1 Digital Signature Transformation (Detached Signatures)

The content to be signed is addressed in the URI= attribute of <ds:Reference>:

```
<ds:Reference URI="myInvoice.xml">
```

An option when using detached digital signatures is to express in XPath that address that qualifies all nodes in the referenced content to be included in the calculation of the digital signature hash. For a signature calculated for a document to remain valid, none of the signed information can change, nor can any information be added or removed from that portion of the document included in the hash calculation.

Consider the need to create a detached signature for a UBL file in which there already exists an enveloped signature. The following transformation element in a digital signature flexibly prevents the signature being invalidated by the subsequent addition of any signatures using the enveloped profile within the extension of the document being signed:

```
<Transform
  Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
 <XPath xmlns:sig=
   "urn:oasis:names:specification:ubl:schema:xsd:CommonSignatureComponents-2">
   count(ancestor-or-self::sig:UBLDocumentSignatures)=0
 </XPath>
</Transform>
```

A non-final transformation algorithm used in the detached signature signs all content outside of any enveloped signatures in the UBL document. When the UBL document does not already have an enveloped signature, one cannot be added without invalidating the detached signature. In effect, the entire document has been signed and cannot change, but the addition of the scaffolding for a signature constitutes a change. However, when the UBL document already has an enveloped signature, other signatures can be added without invalidating the detached signature, because the scaffolding doesn't change when other signatures are added within the existing scaffolding; the non-final transformation algorithm does not include the signatures found in the existing scaffolding. When there is no preexisting enveloped signature, the entire document must be signed in the detached signature.

To sign only a portion of a UBL document, an appropriate [**XPointer**] address SHOULD be used because UBL business object elements do not have attributes of type ID. This requires XPointer awareness on the part of the digital signature tools being used.

5.3 UBL Extension for Enveloped XML Digital Signatures

UBL extensions enable user-defined additions to the standard schemas. The UBL 2.1 schemas in this distribution are provided with a predefined standard extension for enveloped signatures that supports IETF/W3C Digital Signature profiles. These include advanced IETF/W3C XML digital signatures conforming to the ETSI XAdES specification [**XAdES**], thus satisfying EU legal requirements for electronically signed business documents.

This extension also serves as a case study for the creation of user-defined UBL extensions; see Section 5.3.7, "Notes for Extension Creators". Further information on the UBL extension mechanism can be found in [**Customization**].

UBL's implementation of XML digital signatures puts all the signatures relating to a document in a single extension, which is engaged in validation by the `UBL-ExtensionContentDataType-2.1.xsd` schema module.

5.3.1 Namespaces

As is true for the UBL document schemas and common library, the UBL digital signature extension is modeled with three namespaces: one for the apex element (a parallel to the document schema), one for new aggregate constructs (a parallel to the common aggregate schema), and one for new basic constructs (a parallel to the common basic schema). See Figure 63, "UBL Schema Dependencies".

The `urn:oasis:names:specification:ubl:schema:xsd:CommonSignatureComponents-2` namespace is used for the apex element, the `urn:oasis:names:specification:ubl:schema:xsd:SignatureAggregateComponents-2` namespace is used for new aggregate elements, and the `urn:oasis:names:specification:ubl:schema:xsd:SignatureBasicComponents-2` namespace is used for new basic elements. The IETF/W3C digital signature [**xmldsig**] standard namespace `http://www.w3.org/2000/09/xmldsig#` is also used in this extension. These namespaces are bound to the `sig:`, `sac:`, `sbc:` and `ds:` prefixes respectively, but any prefix or even the default namespace can be used for any of these in an XML instance.

Schema fragments for the two XAdES namespaces `http://uri.etsi.org/01903/v1.3.2#` and `http://uri.etsi.org/01903/v1.4.1#` are included and engaged in UBL 2.1 for the convenience of users of the XAdES specification. There is no obligation to use the XAdES extension in the IETF/W3C digital signature.

The table below lists the namespaces used for UBL digital signatures. The prefixes on the left are only documentary conventions; their choice is not constrained by XML.

Table 2. Namespaces for UBL Digital Signatures

Prefix	Namespace	Reference
`ds`	`http://www.w3.org/2000/09/xmldsig#`	[**xmldsig**]
`xades`	`http://uri.etsi.org/01903/v1.3.2#`	[**XAdES**]
`ext`	`urn:oasis:names:specification:ubl:schema:xsd:CommonExtensionComponents-2`	UBL extension namespace
`sig`	`urn:oasis:names:specification:ubl:schema:xsd:CommonSignatureComponents-2`	UBL signature extension apex namespace
`sac`	`urn:oasis:names:specification:ubl:schema:xsd:SignatureAggregateComponents-2`	UBL signature extension aggregate namespace
`sbc`	`urn:oasis:names:specification:ubl:schema:xsd:SignatureBasicComponents-2`	UBL signature extension basic namespace

5.3.2 Identification

This UBL extension is distinguished from other extensions and identified using the URI `urn:oasis:names:specification:ubl:dsig:enveloped` in the `<ext:ExtensionURI>` element.

Note

In addition to Enveloped signatures, Section 5.2.2, "Detached XML Signatures for UBL Documents" also provides methods to be used with Detached signatures (i.e., digital signatures that stand outside the document being signed). Detached signatures constitute an independent technique without associated UBL artefacts, but an example instance showing detached signatures is included in this package; see Section 5.4, "Digital Signature Examples".

5.3.3 Extension Validation

The `UBL-ExtensionContentDataType-2.1.xsd` module links UBL validation to all needed extensions by importing the apex schema fragment of each extension vocabulary. The distribution version of this module supports IETF/W3C XML digital signatures by declaring that the `<ext:ExtensionContent>` element can contain elements from the UBL Digital Signature extension namespace. Accordingly, a single `<sig:UBLDocumentSignatures>` element is used as the apex of all the document's electronic signatures.

The `<ext:ExtensionContent>` element alternatively allows any other namespace apex element in order to allow other foreign extensions in the same document.

5.3.4 Structure

The signature extension structure exists to contain one or more IETF/W3C standard digital signature constructs. The UBL scaffolding for this extension starts with a `<ext:UBLExtension>` element with two children: `<ext:ExtensionURI>` (for extension distinction and identification) and `<ext:ExtensionContent>` (for containing the extension information, in this case the actual signatures and supporting information).

The signature extension Business Information Entities for UBL 2.1 are contained in a single spreadsheet, provided here in two different formats.

`mod/common/UBL-CommonSignatureComponents-2.1.ods`
`mod/common/UBL-CommonSignatureComponents-2.1.xls`

One or more signature extensions in a given document may each contain one or more sets of signature information. The standard scaffolding for a given signature extension begins with the `<ext:UBLExtension>` element. The extension's role as a UBL signature extension is indicated with a child `<ext:ExtensionURI>` element with the `urn:oasis:names:specification:ubl:dsig:enveloped` value. The `urn:oasis:names:specification:ubl:dsig:enveloped:xades` value MAY be used to indicate the use of XAdES in the extension. Other extension metadata elements defined in UBL are allowed to be included for the convenience of users without changing the meaning or use of the extension.

```
<ext:UBLExtension>
  <ext:ExtensionURI
    >urn:oasis:names:specification:ubl:dsig:enveloped</ext:ExtensionURI>
  <ext:ExtensionContent>
```

All uses of the optional `<cbc:ID>` metadata SHOULD be unique so that each extension can be uniquely identified. For the convenience of users, a URI with the URN beginning with `urn:oasis:names:specification:ubl:signature:` and ending with a number value is reserved for this purpose for UBL users, and MAY be used. The value `urn:oasis:names:specification:ubl:signature:3` is a suitable example.

The mandatory <ext:ExtensionContent> element is the extension scaffolding that contains the UBL signature scaffolding. The apex element of the UBL signature information is <sig:UBLDocumentSignatures>. Each <sac:SignatureInformation> aggregate is used to contain the information related to a single IETF/W3C digital signature. Every signature added to the extension is isolated under a separate <sac:SignatureInformation> aggregate element, containing the signature and its supporting information. As many of these aggregates can be in the extension as is needed, each one containing the information for a single digital extension.

```
  <ext:ExtensionContent>
    <sig:UBLDocumentSignatures>
    <sac:SignatureInformation>
    <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
```

Note that three namespaces are used for signature information, in parallel with the UBL design of having a document namespace, aggregate namespace and basic namespace. The apex element is in the urn:oasis:names:specification:ubl:schema:xsd:CommonSignatureComponents-2 namespace, a parallel to a UBL document namespace. Signature-related aggregate entities are in the urn:oasis:names:specification:ubl:schema:xsd:SignatureAggregateComponents-2 namespace. Signature-related basic entities are in the urn:oasis:names:specification:ubl:schema:xsd:SignatureBasicComponents-2 namespace. Accordingly, there are three W3C Schema fragments in the distribution accommodating these three namespaces.

An aggregate MAY be identified for referencing purposes using the common library <cbc:ID> element. Such an identifier may be useful in workflow scenarios where a particular signature needs to be identified external to the document, but its use is not obligatory. The identifier used can be any value, but for convenience the URI of a URN beginning with urn:oasis:names:specification:ubl:signatures: and ending with a number value is reserved for this purpose for UBL users. An example is urn:oasis:names:specification:ubl:signatures:3. As with all identifiers, each SHOULD be unique across all identifier values in a given UBL instance.

An aggregate MAY make reference to an existing <cac:Signature> business object in the same UBL document, but this is not obligatory. When needed, the <sbc:ReferencedSignatureID> basic element is used to point to the <cbc:ID> identifier value of the referenced <cac:Signature>. The identifier used can be any value, but for convenience the URI of a URN beginning with urn:oasis:names:specification:ubl:signatures: and ending with the local name of the parent of the signature business object and optionally followed with a colon and number, as in the urn:oasis:names:specification:ubl:signatures:IssuerEndorsement example, is reserved for this purpose for UBL users. As with all identifier references, the referenced identifier should exist and should be unique across all such identifier values in a given UBL instance.

A single <ds:Signature> element is a child of the aggregate. It MAY be absent from the document, thus supporting workflow scenarios where the element is added by a subsequent process after the UBL scaffolding is added by an earlier process. However, the signature information is semantically incomplete without the IETF/W3C-defined element. To support countersignatures countersigning this signature, this element must use the Id= attribute with a value unique from other attributes of schema type ID in the instance.

A skeleton example of a single signature is as follows:

```
<ext:ExtensionContent>
  <sig:UBLDocumentSignatures
    xmlns:sig=
      "urn:oasis:names:specification:ubl:schema:xsd:CommonSignatureComponents-2"
    xmlns:sac=
      "urn:oasis:names:specification:ubl:schema:xsd:SignatureAggregateComponents-2"
    xmlns:sbc=
      "urn:oasis:names:specification:ubl:schema:xsd:SignatureBasicComponents-2">
    <sac:SignatureInformation>
```

```
      <cbc:ID>urn:oasis:names:specification:ubl:signature:1</cbc:ID>
      <sbc:ReferencedSignatureID
        >urn:oasis:names:specification:ubl:signature:Invoice
      </sbc:ReferencedSignatureID>
      <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#" Id=...>
        <ds:SignedInfo>
         ...
          <ds:Reference URI=...>
           ...
            <ds:Transform>
             ...
            </ds:Transform>
           ...
          </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>
          ...
        </ds:SignatureValue>
        <ds:KeyInfo>
         ...
        </ds:KeyInfo>
        <ds:Object>
         ...
        </ds:Object>
      </ds:Signature>
    </sac:SignatureInformation>
  </sig:UBLDocumentSignatures>
</ext:ExtensionContent>
```

Note

The XAdES specification contains all qualifying XAdES information in a single `<ds:Object>` element located as shown above. The UBL 2.1 distribution includes and engages XAdES schema fragments versioned 1.3.2 and 1.4.1 for the convenience of users who choose to use these versions of XAdES. Users of the UBL signature extension are not obliged to use any XAdES extensions.

See Section 5.2.1, “Enveloped XML Signatures in UBL Documents” for rules regarding UBL signature elements in the unextended portion of UBL documents and an example of their use.

5.3.5 Transformation

The content to be signed is indicated in the `URI=` attribute of `<ds:Reference>`. Using the empty string indicates that the entire document (i.e. the enveloping UBL instance) is what is being signed:

```
<ds:Reference URI="">
```

A requirement when using digital signatures is to express in XPath that address that qualifies all nodes in the referenced content to be included in the calculation of the digital signature hash. For a signature added to a document to remain valid, none of the information can change, nor can any information be added or removed from that portion of the document included in the hash calculation.

One of two such transformation expressions SHOULD be used in the UBL signature extension; users should choose the appropriate one to meet the objectives of the signature being added to the document. Adding non-signature information to the UBL document will invalidate all signatures already in the extension. The choice to make is whether to support additional signatures after adding the signature with the transformation expression.

The following transformation element in a digital signature flexibly prevents the signature from being invalidated by the subsequent addition of other signatures within the extension:

```
<Transform
     Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
   <XPath xmlns:sig=
     "urn:oasis:names:specification:ubl:schema:xsd:CommonSignatureComponents-2">
     count(ancestor-or-self::sig:UBLDocumentSignatures |
       here()/ancestor::sig:UBLDocumentSignatures[1]) >
         count(ancestor-or-self::sig:UBLDocumentSignatures)
   </XPath>
 </Transform>
```

The following transformation element in a digital signature is inflexible and thus would be considered a “final” signature to be added to the document. Such a signature will be invalidated by the subsequent addition of other signatures to the document:

```
<Transform
     Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
   <XPath xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
     count(ancestor-or-self::ds:Signature |
       here()/ancestor::ds:Signature[1]) >
         count(ancestor-or-self::ds:Signature)
   </XPath>
 </Transform>
```

Multiple separate items of extra-document content (e.g., attachments) or embedded W3C signature content can be included in the same signature by using sibling `<ds:Reference>` elements with other `URI=` attribute values. For example, to countersign another signature in the same UBL document, make a local reference to that signature’s unique identifier, as in:

```
<ds:Reference URI="#{Id attribute of ds:Signature}">
```

Note

To digitally sign only a portion of standard UBL content and not the entire document of UBL content, one uses an appropriate XPointer address for `URI=`. This requires XPointer awareness on the part of the digital signature tools being used.

5.3.6 Extension Validation Methodology

The single extension built into the UBL 2.1 distribution validates transparently, and the UBL extension mechanism allows the addition of other extensions in the same instance.

Users wishing to validate other extensions found in the instance simply revise the `UBL-ExtensionContentDataType-2.1.xsd` schema fragment. An `<xsd:import>` directive is added to incorporate the schema constraints of the apex of another extension to be validated in the single pass of XSD validation. Figure 63, “UBL Schema Dependencies” shows the replacement of the schema fragment with one in which user-defined extension modules with namespaces `ext:`, `zzz:`, `zac:`, and `zbc:` augment the digital signature extension modules with namespaces `ext:`, `sig:`, `sac:`, `sbc:` and `ds:`.

Due to limitations of W3C Schema validation semantics (this is not the case in RELAX NG, for example), the apex element of the extension in the instance being validated cannot be constrained solely to the apex element declared. The lax validation permits any element declared in any schema fragment to be the apex of an extension. Thus, an instance will pass when a known extension element not permitted by the user to be an apex element is in the place of an apex element. This is simply regarded by downstream processes as an unknown extension and will likely be ignored.

5.3.7 Notes for Extension Creators

The UBL Digital Signature extension has been modeled as an example to follow when designing and writing other custom extensions. The following points should be noted:

- Extension designers should follow the example in providing separate namespaces for apex element, aggregate constructs, and basic constructs if they wish the new items to be considered for inclusion in future UBL releases. This structures the new items for inclusion in the UBL common library. See `xml/MyTransportationStatus.xml` for a document instance exemplifying the recommended treatment of namespaces.

- Whenever possible, existing UBL common library aggregate and basic constructs should be used in extensions rather than inventing new items with the same semantics. However, a common library aggregate construct should only be used when the entire aggregate and all of its descendants are applicable in the extension context without any changes. If any items must be removed, then a new extension aggregate with a new local name should be used. If all the constructs are applicable but some items need to be added, then a new extension aggregate with the same local name as the common library aggregate should be used, and the common library aggregate should be copied with the new constructs inserted.

5.4 Digital Signature Examples

The `xml/UBL-Invoice-2.0-Enveloped.xml` sample document illustrates the embedding of three extensions in a single document, one of which is a bona fide verifiable enveloped signature extension. A `<cac:Signature>` element makes reference to the embedded signature.

The `xml/UBL-Invoice-2.0-Detached.xml` sample document illustrates the detaching of a digital signature outside of the UBL file. A `<cac:Signature>` element makes reference to the external signature.

The `xml/UBL-Invoice-2.0-Detached-Signature.xml` instance is a bona fide verifiable digital signature of the `xml/UBL-Invoice-2.0-Detached.xml` instance.

6 Conformance

6.1 Document and Schema Conformance

Conformance as applied to UBL documents and schemas, and the distinction between UBL conformance and UBL compatibility is described in detail in the *UBL 2 Guidelines for Customization*[**Customization**].

6.2 Digital Signature Conformance

Claiming syntax conformance to the enveloped signature profile of this specification requires:

- the schema-valid expression of a UBL extension when the UBL Signature apex element is the apex of the extension;
- the <ext:Extension> element is present in the UBL extension and has either urn:oasis:names:specification:ubl:dsig:enveloped or urn:oasis:names:specification:ubl:dsig:enveloped:xades as its value;
- the value in all uses of <sbc:ReferencedSignatureID>, when present, correlates to a corresponding <cbc:ID> element of a <cac:Signature> element in the same instance; and
- the <cbc:SignatureMethod> element, when present, of signature business objects whose signatures are in the UBL extension has either urn:oasis:names:specification:ubl:dsig:enveloped or urn:oasis:names:specification:ubl:dsig:enveloped:xades as its value.

Claiming processing conformance to the enveloped profile of this specification requires the conformant processing of all contained <ds:Signature> elements per [**xmldsig**].

Claiming syntax conformance to the detached profile of this specification requires that the <cbc:SignatureMethod> element, when present, of signature business objects whose signatures are outside of the UBL document has either urn:oasis:names:specification:ubl:dsig:detached or urn:oasis:names:specification:ubl:dsig:detached:xades as its value.

6.3 XAdES Conformance

When conformance to XAdES in a UBL document is chosen, this specification requires the valid expression and processing of the XAdES syntax found in an XMLDSig per [**XAdES**].

Appendix A Release Notes (Non-Normative)

A.1 Availability

Online and downloadable versions of this release are available from the locations specified at the top of this document.

A.2 Status of this Release

Release of this package to the public begins its third public review. The UBL Technical Committee actively solicits input from the user community regarding this release. See Status at the beginning of this document for procedures to be used in submitting comments to the Committee. Note that in accordance with OASIS policies regarding intellectual property, the UBL TC *cannot* accept input from persons outside the UBL TC (including OASIS members) unless it is submitted via the comment list.

THIS RELEASE IS SUBJECT TO CHANGE. IT IS PROVIDED FOR TESTING PURPOSES ONLY AND SHOULD NOT BE USED FOR PRODUCTION SYSTEMS.

A.3 Package Structure

The third public review draft of the UBL 2.1 specification is published as a zip archive named prd3-UBL-2.1.zip. Unzipping this archive creates a directory named prd3-UBL-2.1 containing a master DocBook XML file (UBL-2.1.xml), a generated hypertext version of this file (UBL-2.1.html), a generated PDF version of this file (UBL-2.1.pdf), and a number of subdirectories. The files in these subdirectories, linked to from UBL-2.1.xml, UBL-2.1.html, and UBL-2.1.pdf, contain the various normative and informational pieces of the 2.1 release. A description of each subdirectory is given below. Note that while the UBL-2.1.xml file is the “original” of this specification, it may not be viewable in all currently available web browsers.

`art`
: Diagrams and illustrations used in this specification

`asn`
: ASN.1 UBL 2.1 schema; see Section H.1, “ASN.1 UBL 2.1 Specification”

`cl`
: Code list specification files; see Appendix F, *UBL 2.1 Code Lists and Two-phase Validation (Non-Normative)*

`css`
: CSS stylesheets for viewing UBL-2.1.html

`cva`
: Artefacts expressing data type qualifications; see [**CVA**] in Section 1.2, “Normative References” and Figure E.1, “Data Type Qualification in UBL 2.1” in Appendix E, *Data Type Qualifications in UBL (Non-Normative)*

`db`
: DocBook stylesheets for viewing UBL-2.1.xml

`etc`
: Miscellaneous supporting information

`mod`
: Spreadsheet data models; see Appendix C, *The UBL 2.1 Data Model (Non-Normative)*

rnc
Alternative versions of the UBL 2.1 schemas in RELAX NG (compact syntax); see Section H.2, “UBL 2.1 RELAX NG Schemas”

val
Test harness for demonstrating UBL 2.1 two-phase validation; see Appendix F, *UBL 2.1 Code Lists and Two-phase Validation (Non-Normative)*

xml
Sample UBL 2.1 instances; see Appendix G, *UBL 2.1 Example Document Instances (Non-Normative)*

xsd
XSD schemas; see Section 3, “UBL 2.1 Schemas”

xsdrt
“Runtime” XSD schemas; see Section 3, “UBL 2.1 Schemas”

A.4 Support

UBL is a volunteer project of the international business community. Inquiries regarding UBL may be posted to the public ubl-dev list, archives for which are located at

http://lists.oasis-open.org/archives/ubl-dev/

Subscriptions to ubl-dev can be made through the OASIS list manager at

http://www.oasis-open.org/mlmanage/index.php

OASIS provides an official community gathering place and information resource for UBL at

http://ubl.xml.org/

A.5 UBL Customization

UBL provides a vocabulary that, for many user communities, can be used “as is”. However, it is recognized that some user communities must address use cases whose requirements are not met by the UBL off-the-shelf solution. A separate OASIS Committee Specification known as the *UBL 2 Guidelines for Customization*[**Customization**] has been published to aid such users in developing custom solutions based on UBL.

The goal of UBL customization is to maintain a common understanding of the meaning of information being exchanged between specific implementations. The factors governing when to customize may be business-driven, technically driven, or both. The decision should be based on real-world needs balanced against perceived economic benefits.

A.6 Upgrading from UBL 2.0 to UBL 2.1

For current UBL implementers, the most important thing to know about UBL 2.1 is that it is completely backward-compatible with UBL 2.0. In other words, any document that validates against a UBL 2.0 schema will validate against the UBL 2.1 version of that schema. The remaining differences relate mainly to functionality that has been added to the 2.0 framework in the areas of eTendering, sales reporting, utility statements, transport handling, and collaborative planning, forecasting, and replenishment (CPFR®).

Nonetheless, it would be unwise to simply overlay this UBL 2.1 release onto an existing 2.0 installation, and the possible differences among existing 1.0 and 2.0 installations are too large to allow a specific set of instructions to be provided for making the transition.

The brief history of UBL document types in the next section puts the new capabilities into context and may help owners of existing UBL 1.0 and 2.0 installations decide whether to upgrade to 2.1. New 2.1 users, on the other hand, can simply install 2.1 and rest assured that their software will interoperate with UBL documents generated by existing conformant UBL 2.0 installations. For more on the concept of conformance, see [**Customization**].

A.7 Dictionary Entry Name Corrections in UBL 2.1

Dictionary Entry Names (DENs) uniquely identify every BIE in the UBL library using the methodology described in [**CCTS**]. Several errors in dictionary entry naming were discovered and corrected in the course of preparing UBL 2.1. These corrections have no effect on processing, validation, or instance generation, but they are listed in Section B.3.4, “Changes from UBL 2.0 XML to UBL 2.1 XML” for completeness in documentation.

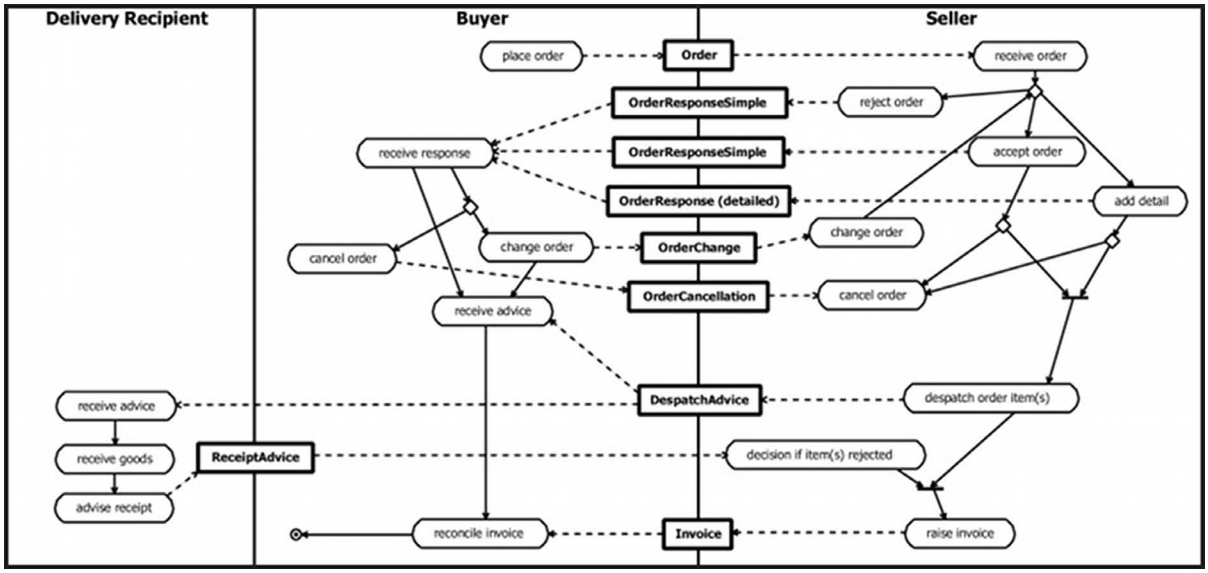
Appendix B Revision History (Non-Normative)

Since its first release as an OASIS Standard in 2004, UBL has experienced one major and one minor version upgrade.

B.1 UBL 1.0

Though apparently limited in scope, the eight document types provided in UBL 1.0 (2004) are applicable to a very large number of real-world use cases and have been widely deployed. These original 1.0 document types, later updated in UBL 2.0 and continued here in 2.1, are Order, Order Response, Order Response Simple, Order Change, Order Cancellation, Despatch Advice, Receipt Advice, and Invoice. The figure below shows the original assumed process context for this most basic set of UBL document types. The scope of the process corresponds roughly to that of the UBL 2 Order, Fulfillment, and Traditional Billing processes described in the text (see Section 2.6, “Ordering”, Section 2.7, “Fulfilment”, and Section 2.8.2, “Traditional Billing”).

Figure B.1. UBL 1.0 Order-to-Invoice Business Process



Because versions of UBL beginning with 2.0 do not maintain backward compatibility with UBL 1.0 document instances (that is, UBL 1.0 document instances will not validate against schemas from UBL 2.0 and later), use of UBL 1.0 in new installations is deprecated. Suitably revised versions of the original eight document types continue all the business functionality of UBL 1.0 in later versions.

B.2 Major Revision: UBL 2.0

Adoption of UBL 1.0 following ratification as an OASIS standard in November 2004 resulted in major inputs of new business content beyond the eight basic order-to-invoice business documents specified in the original release. In particular, contributions from representatives of government procurement, taxation, and transportation agencies in Europe, Asia, and North America resulted in greatly expanded pre-order and post-invoice capabilities together with the addition of several transport-related document types, bringing the total number of document types in UBL 2.0 to 31.

The new release also featured changes in UBL’s use of XML schema methodology—most importantly, the adoption of global scoping for all element types—breaking backward compatibility with UBL 1.0 instances and therefore necessitating designation as a major revision, signified by incrementing the version number from 1.0 to 2.0 rather than 1.1. The original eight UBL 1.0 document types were revised to reflect these changes.

UBL 2.0 achieved OASIS Standardization in December 2006, and the package was updated and corrected in May 2008.

The 23 document types added in UBL 2.0 can be summarized as follows:

Added UBL 2.0 document types for sourcing: Catalogue, Catalogue Deletion, Catalogue Item Specification Update, Catalogue Pricing Update, Catalogue Request, Quotation, Request for Quotation

Added UBL 2.0 document types for fulfilment: Bill of Lading, Certificate of Origin, Forwarding Instructions, Packing List, Transportation Status, Waybill

Added UBL 2.0 document types for billing: Credit Note, Debit Note, Freight Invoice, Reminder, Self Billed Credit Note, Self Billed Invoice

Added UBL 2.0 document types for payment: Remittance Advice, Statement

Added UBL 2.0 supplementary document types: Application Response, Attached Document

B.3 Minor Revision: UBL 2.1

B.3.1 New Document Types in UBL 2.1

Because it preserves backward compatibility with UBL 2.0, UBL 2.1 is technically a minor release, not a major one. However, it does add 36 new document types, bringing the total number of UBL business documents to 67.

Added UBL 2.1 document types for eTendering: Awarded Notification, Call for Tenders, Contract Award Notice, Contract Notice, Guarantee Certificate, Tender, Tender Receipt, Tenderer Qualification, Tenderer Qualification Response, Unawarded Notification

Added UBL 2.1 document types for Collaborative planning, forecasting, and replenishment: Exception Criteria, Exception Notification, Forecast, Forecast Revision, Item Information Request, Prior Information Notice, Trade Item Location Profile

Added UBL 2.1 document types for Vendor Managed Inventory: Instruction for Returns, Inventory Report, Performance History, Product Activity, Retail Event, Stock Availability Report

Added UBL 2.1 document types for Intermodal Freight Management: Goods Item Itinerary, Transport Execution Plan, Transport Execution Plan Request, Transport Progress Status, Transport Progress Status Request, Transport Service Description, Transport Service Description Request, Transportation Status, Transportation Status Request

Added UBL 2.1 document type for Utility billing: Utility Statement

Added UBL 2.1 supplementary document types: Despatch Cancellation, Document Status, Document Status Request

B.3.2 Financial Information Enhancements in UBL 2.1

UBL 2.1 has been enhanced to support the financial information required for downstream processing of Invoices within financial services. Through standardization, business vocabularies such as UBL for eBusiness and ISO 20022 for eFinance enable Straight Through Processing (STP) and paperless trading along the entire Financial Supply Chain.

Based on analysis conducted during the current development cycle by the UBL Financial Information Requirements Task Group (FIRTG), the following enhancements have been included in UBL 2.1:

Financial account: Aligned with today's needs and designed for truly global usage (`AliasName`, `AccountTypeCode`, ...). A financial account can now be associated to the Person information aggregate, not only to a Party.

Payment mandate information can optionally be sent as part of the Order; this can be considered a simplification for small businesses.

Trade financing: UBL 2.1 is designed to support basic trade financing practices (invoice financing, factoring, pre-shipment/order financing, Letter of Credit, ...)

Payments reconciliation: UBL Invoice and Remittance Advice can be used together with financial messages to ensure end-to-end transport of reconciliation identifiers (invoicing party references). In particular, UBL provides a solution for advanced external remittance, where the UBL Remittance Advice is used to transmit the details of complex remittance information associated with the payment initiation process (see ISO 20022 guides for details). Person is now enriched with a person identification, which is often required by the banking sector for legal reasons.

Currency Amounts: UBL 2.1 features improved handling of alternative currency amounts.

UBL 2.1 also includes enhancements to legal information.

Party Legal Entity: The Party's legal information has been considerably enriched with information required by advanced procurement and global usage.

Service Provider Party: The electronic trade is increasingly supported and executed through Service Providers into several forms like the outsourcing and ASP modes.

UBL Party is now improved to keep track of services handled by one or more service providers.

Power of Attorney can now be associated with a Party.

B.3.3 Revised Approach to Data Definitions in UBL 2.1

Since the definitions of the 4000+ uniquely named data items in UBL 2.1 are normative in themselves, it is important that each be specified with the greatest care possible. However, there are two very different approaches that can be taken in accomplishing this objective.

One approach, used more or less by default in UBL 2.0, is to give a definition for each item that correctly describes that item as a node in a data model. This is the approach that resulted in (for example) the UBL 2.0 definition of `TaxTotal` in `CreditNoteLine` as "An association to Tax Total". While perfectly correct from the standpoint of data model construction (the ASBIE for `TaxTotal` is, in fact, an association to the class or ABIE named Tax Total; see Appendix C, *The UBL 2.1 Data Model (Non-Normative)*), such a definition is less than ideally useful for the business analyst attempting to determine the meaning of `TaxTotal` as an element name in an actual Credit Note. Of course, the definition of the ABIE itself could be tracked back through this reference to its place in the model, where in UBL 2.0 one would find the class "Tax Total" defined as "Information about a total amount of a particular tax", but even here an attempt to be absolutely correct can lead to a pathological expansion of the definition, as "Information about a total amount of a particular tax" becomes "A description of information about a total amount of a particular tax" and then "Structured data constituting a description of information about a total amount of a particular tax" and so on.

A second approach to wording the definitions, adopted here in UBL 2.1, is to write each definition as an answer to the question "What does this mean?" in the context of an actual document instance containing

the item. Thus, in UBL 2.1, the definition of `TaxTotal` in `CreditNoteLine` has changed from “An association to Tax Total” to “A total amount of taxes of a particular kind applicable to this credit note line”, which the designers of UBL 2.1 believe to be more directly usable to someone analyzing an instance of a Credit Note. This compression of levels of abstraction has been carried through so far as to eliminate phrases such as “A description of” in the great majority of cases, leaving it to the reader's common sense to understand that a definition such as “The delivery requested by the party requesting a transportation service” refers to a description of the act of delivery and not the physical delivery itself, which of course could never literally be part of a document. The only widespread exceptions to this policy are the definitions of ABIEs in the Common Library, which begin “A class to describe...” or sometimes “A class to define...” in order to emphasize their more abstract role. The designers of UBL 2.1 believe that anything lost in technical correctness through this approach will be more than compensated for in improved practical usability.

B.3.4 Changes from UBL 2.0 XML to UBL 2.1 XML

The following three tables show the differences between the XML elements and attributes in UBL 2.0 (as updated in 2008) and those in UBL 2.1.

All changes in 2.1 schemas are backward-compatible with valid UBL 2.0 instances. Changes include the addition of new elements and attributes; changes in cardinality from 1 to 0..1 (i.e., making a formerly required element optional); changes in cardinality from 0..1 to 0..n (i.e., allowing an unlimited number of occurrences instead of just one); and corrections to Dictionary Entry Names (DENs).

B.3.4.1 Changes to Library Elements, UBL 2.0 to UBL 2.1

The following table sums up the differences between the XML elements in the UBL 2.0 Common Library (as updated in 2008) and those in the UBL 2.1 Common Library.

Table B.1. Changes to Library Elements, UBL 2.0 to UBL 2.1

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
ActivityDataLine		Added
ActivityProperty		Added
Address		
	LocationCoordinate	Changed cardinality from 0..1 to 0..n
AllowanceCharge		
	PerUnitAmount	Added
AppealTerms		Added
AuctionTerms		Added
AwardingCriteria		Added
AwardingCriteriaResponse		Added
AwardingTerms		Added
BudgetAccount		Added
BudgetAccountLine		Added
BudgetAmount		Added
Capability		Added
CatalogueLine		
	ReplacedRelatedItem	Added
	KeywordItemProperty	Added
Certificate		Added
CertificateOfOriginApplication		
	ExporterParty	Added
	ImporterParty	Added
ClassificationScheme		
	Note	Changed cardinality from 0..1 to 0..n
Clause		Added
CompletedTask		Added
Consignment		
	CarrierAssignedID	Added
	ConsigneeAssignedID	Added
	ConsignorAssignedID	Added
	FreightForwarderAssignedID	Added
	BrokerAssignedID	Added
	ContractedCarrierAssignedID	Added
	PerformingCarrierAssignedID	Added
	AnimalFoodIndicator	Added
	HumanFoodIndicator	Added
	LivestockIndicator	Added
	BulkCargoIndicator	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	ContainerizedIndicator	Added
	GeneralCargoIndicator	Added
	SpecialSecurityIndicator	Added
	ThirdPartyPayerIndicator	Added
	CarrierServiceInstructions	Added
	CustomsClearanceServiceInstructions	Added
	ForwarderServiceInstructions	Added
	SpecialServiceInstructions	Added
	SequenceID	Added
	ShippingPriorityLevelCode	Added
	HandlingCode	Added
	HandlingInstructions	Added
	Information	Added
	TotalGoodsItemQuantity	Added
	TotalTransportHandlingUnitQuantity	Added
	InsuranceValueAmount	Added
	DeclaredForCarriageValueAmount	Added
	DeclaredStatisticsValueAmount	Added
	FreeOnBoardValueAmount	Added
	SpecialInstructions	Added
	SplitConsignmentIndicator	Added
	DeliveryInstructions	Added
	ConsignmentQuantity	Added
	ConsolidatableIndicator	Added
	HaulageInstructions	Added
	LoadingSequenceID	Added
	CustomsIdentification	Added
	RequestedPickupTransportEvent	Added
	RequestedDeliveryTransportEvent	Added
	PlannedPickupTransportEvent	Added
	PlannedDeliveryTransportEvent	Added
	Status	Added
	ChildConsignment	Added
	PerformingCarrierParty	Added
	SubstituteCarrierParty	Added
	LogisticsOperatorParty	Added
	TransportAdvisorParty	Added
	HazardousItemNotificationParty	Added
	InsuranceParty	Added
	MortgageHolderParty	Added
	BillOfLadingHolderParty	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	CollectPaymentTerms	Added
	DisbursementPaymentTerms	Added
	PrepaidPaymentTerms	Added
	ExtraAllowanceCharge	Added
	MainCarriageShipmentStage	Added
	PreCarriageShipmentStage	Added
	OnCarriageShipmentStage	Added
	TransportHandlingUnit	Added
	FirstArrivalPortLocation	Added
	LastExitPortLocation	Added
	ConsolidatedShipment	Added
Consumption		Added
ConsumptionAverage		Added
ConsumptionHistory		Added
ConsumptionLine		Added
ConsumptionPoint		Added
ConsumptionReport		Added
ConsumptionReportReference		Added
Contact		
	Note	Changed cardinality from 0..1 to 0..n
Contract		
	NominationDate	Added
	NominationTime	Added
	Note	Added
	VersionID	Added
	Description	Added
	NominationPeriod	Added
	ContractualDelivery	Added
ContractExecutionRequirement		Added
ContractExtension		Added
ContractingParty		Added
Correction		Added
CreditNoteLine		
	Note	Changed cardinality from 0..1 to 0..n
	PaymentPurposeCode	Added
	FreeOfChargeIndicator	Added
	InvoicePeriod	Added
	OrderLineReference	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	OriginatorParty	Added
	PaymentTerms	Added
	AllowanceCharge	Added
	DeliveryTerms	Added
	SubCreditNoteLine	Added
CustomsIdentification		Added
DebitNoteLine		
	Note	Changed cardinality from 0..1 to 0..n
	PaymentPurposeCode	Added
	AllowanceCharge	Added
	SubDebitNoteLine	Added
Declaration		Added
Delivery		
	ReleaseID	Added
	AlternativeDeliveryLocation	Added
	CarrierParty	Added
	NotifyParty	Added
	DeliveryTerms	Added
	MinimumDeliveryUnit	Added
	MaximumDeliveryUnit	Added
	Shipment	Added
DeliveryTerms		
	Amount	Added
DependentPriceReference		Added
Despatch		
	GuaranteedDespatchDate	Added
	GuaranteedDespatchTime	Added
	ReleaseID	Added
	Instructions	Added
	DespatchLocation	Added
	CarrierParty	Added
	NotifyParty	Added
	EstimatedDespatchPeriod	Added
	RequestedDespatchPeriod	Added
DespatchLine		
	Note	Changed cardinality from 0..1 to 0..n
DocumentReference		
	IssueTime	Added
	LanguageID	Added
	LocaleCode	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	VersionID	Added
	DocumentStatusCode	Added
	DocumentDescription	Added
	ValidityPeriod	Added
	IssuerParty	Added
	ResultOfVerification	Added
DocumentResponse		
	DocumentReference	Changed cardinality from 1 to 1..n
Duty		Added
EconomicOperatorShortList		Added
EmissionCalculationMethod		Added
EnergyTaxReport		Added
EnergyWaterSupply		Added
EnvironmentalEmission		Added
EvaluationCriteria		Added
Event		Added
EventComment		Added
EventLineItem		Added
EventTactic		Added
EventTacticEnumeration		Added
Evidence		Added
ExceptionCriteriaLine		Added
ExceptionNotificationLine		Added
ExternalReference		
	HashAlgorithmMethod	Added
	MimeCode	Added
	FormatCode	Added
	EncodingCode	Added
	CharacterSetCode	Added
	FileName	Added
	Description	Added
FinancialAccount		
	AliasName	Added
	AccountFormatCode	Added
FinancialGuarantee		Added
ForecastException		Added
ForecastExceptionCriteriaLine		Added
ForecastLine		Added
ForecastRevisionLine		Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
FrameworkAgreement		Added
GoodsItem		
	ID	Changed cardinality from 1 to 0..1
	ChargeableQuantity	Added
	ReturnableQuantity	Added
	TraceID	Added
	Delivery	Added
	Pickup	Added
	Despatch	Added
	MeasurementDimension	Added
	ContainingPackage	Added
	ShipmentDocumentReference	Added
ImmobilizedSecurity		Added
InstructionForReturnsLine		Added
InventoryReportLine		Added
InvoiceLine		
	Note	Changed cardinality from 0..1 to 0..n
	PaymentPurposeCode	Added
	InvoicePeriod	Added
	WithholdingTaxTotal	Added
	SubInvoiceLine	Added
Item		
	Certificate	Added
	Dimension	Added
ItemComparison		
	PriceAmount	Changed dictionary entry name from "Item Comparison. Price. Amount" to "Item Comparison. Price Amount. Amount"
ItemIdentification		
	BarcodeSymbologyID	Added
ItemInformationRequest-Line		Added
ItemInstance		
	BestBeforeDate	Added
ItemLocationQuantity		
	Package	Added
	AllowanceCharge	Added
	DependentPriceReference	Added
ItemManagementProfile		Added
ItemProperty		
	ID	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	NameCode	Added
	TestMethod	Added
	Value	Changed cardinality from 1 to 0..1
	ValueQuantity	Added
	ValueQualifier	Added
	ImportanceCode	Added
	ListValue	Added
	RangeDimension	Added
	ItemPropertyRange	Added
ItemPropertyGroup		
	ImportanceCode	Added
ItemPropertyRange		Added
LineItem		
	Note	Changed cardinality from 0..1 to 0..n
	WarrantyInformation	Added
	SubLineItem	Added
	WarrantyValidityPeriod	Added
	WarrantyParty	Added
Location		
	Description	Changed cardinality from 0..1 to 0..n
	LocationTypeCode	Added
	InformationURI	Added
	Name	Added
	SubsidiaryLocation	Added
	LocationCoordinate	Added
LocationCoordinate		
	AltitudeMeasure	Added
MaritimeTransport		
	RadioCallSignCode	Added
Meter		Added
MeterProperty		Added
MeterReading		Added
MiscellaneousEvent		Added
MonetaryTotal		
	AllowanceTotalAmount	Changed dictionary entry name from "Monetary Total. Allowance Total Amount. Amount" to "Monetary Total. Allowance_ Total Amount. Amount"

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	ChargeTotalAmount	Changed dictionary entry name from "Monetary Total. Charge Total Amount. Amount" to "Monetary Total. Charge_ Total Amount. Amount"
	PayableAlternativeAmount	Added
NotificationRequirement		Added
OnAccountPayment		Added
OrderLine		
	Note	Changed cardinality from 0..1 to 0..n
	OrderLineReference	Added
OrderReference		
	SalesOrderID	Changed dictionary entry name from "Order Reference. Sales Order Identifier. Identifier" to "Order Reference. Sales_ Order Identifier. Identifier"
	OrderTypeCode	Added
Package		
	TraceID	Added
	ContainingTransportEquipment	Added
	Delivery	Added
	Pickup	Added
	Despatch	Added
Party		
	IndustryClassificationCode	Added
	Person	Changed cardinality from 0..1 to 0..n
	ServiceProviderParty	Added
	PowerOfAttorney	Added
	FinancialAccount	Added
PartyLegalEntity		
	RegistrationDate	Added
	RegistrationExpirationDate	Added
	CompanyLegalFormCode	Added
	SoleProprietorshipIndicator	Added
	CompanyLiquidationStatusCode	Added
	CorporateStockAmount	Added
	FullyPaidSharesIndicator	Added
	HeadParty	Added
	ShareholderParty	Added
PaymentMandate		Added
PaymentMeans		

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	PaymentMandate	Added
	TradeFinancing	Added
PaymentTerms		
	PaymentMeansID	Changed cardinality from 0..1 to 0..n
	PaymentPercent	Added
	SettlementDiscountAmount	Added
	PenaltyAmount	Added
	PaymentTermsDetailsURI	Added
	PaymentDueDate	Added
	InstallmentDueDate	Added
	InvoicingPartyReference	Added
	ExchangeRate	Added
	ValidityPeriod	Added
PerformanceDataLine		Added
Person		
	ID	Added
	OtherName	Added
	NationalityID	Added
	GenderCode	Added
	BirthDate	Added
	BirthplaceName	Added
	Contact	Added
	FinancialAccount	Added
	IdentityDocumentReference	Added
	ResidenceAddress	Added
Pickup		Added
PowerOfAttorney		Added
Price		
	PricingExchangeRate	Added
ProcessJustification		Added
ProcurementProject		Added
ProcurementProjectLot		Added
ProjectReference		Added
PromotionalEvent		Added
PromotionalEventLineItem		Added
PromotionalSpecification		Added
QualificationResolution		Added
QualifyingParty		Added
QuotationLine		
	Note	Changed cardinality from 0..1 to 0..n

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	RequestForQuotationLineID	Added
	AlternativeLineItem	Added
	RequestLineReference	Added
ReceiptLine		
	Note	Changed cardinality from 0..1 to 0..n
	QuantityDiscrepancyCode	Added
	OversupplyQuantity	Changed dictionary entry name from "Receipt Line. Oversupply Quantity. Quantity" to "Receipt Line. Oversupply_ Quantity. Quantity"
Regulation		Added
ReminderLine		
	Note	Changed cardinality from 0..1 to 0..n
	PenaltySurchargePercent	Added
	Amount	Added
	PaymentPurposeCode	Added
RemittanceAdviceLine		
	Note	Changed cardinality from 0..1 to 0..n
	PaymentPurposeCode	Added
	InvoicingPartyReference	Added
Renewal		Added
RequestForQuotationLine		
	Note	Changed cardinality from 0..1 to 0..n
	OptionalLineItemIndicator	Added
	PrivacyCode	Added
	SecurityClassificationCode	Added
RequestForTenderLine		Added
Response		
	ReferenceID	Changed cardinality from 1 to 0..1
	EffectiveDate	Added
	EffectiveTime	Added
	Status	Added
ResultOfVerification		Added
RetailPlannedImpact		Added
SalesItem		Added
ServiceProviderParty		Added
ShareholderParty		Added
Shipment		
	ConsignmentQuantity	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	Consignment	Changed cardinality from 1 to 0..n
	ReturnAddress	Added
ShipmentStage		
	PreCarriageIndicator	Changed cardinality from 1 to 0..1
	OnCarriageIndicator	Changed cardinality from 1 to 0..1
	EstimatedDeliveryDate	Added
	EstimatedDeliveryTime	Added
	RequiredDeliveryDate	Added
	RequiredDeliveryTime	Added
	LoadingSequenceID	Added
	SuccessiveSequenceID	Added
	Instructions	Added
	DemurrageInstructions	Added
	LoadingTransportEvent	Added
	ExaminationTransportEvent	Added
	AvailabilityTransportEvent	Added
	ExportationTransportEvent	Added
	DischargeTransportEvent	Added
	WarehousingTransportEvent	Added
	TakeoverTransportEvent	Added
	OptionalTakeoverTransportEvent	Added
	DropoffTransportEvent	Added
	ActualPickupTransportEvent	Added
	DeliveryTransportEvent	Added
	ReceiptTransportEvent	Added
	StorageTransportEvent	Added
	AcceptanceTransportEvent	Added
	TerminalOperatorParty	Added
	CustomsAgentParty	Added
	EstimatedTransitPeriod	Added
	FreightAllowanceCharge	Added
	FreightChargeLocation	Added
	DetentionTransportEvent	Added
	RequestedDepartureTransportEvent	Added
	RequestedArrivalTransportEvent	Added
	RequestedWaypointTransportEvent	Added
	PlannedDepartureTransportEvent	Added
	PlannedArrivalTransportEvent	Added
	PlannedWaypointTransportEvent	Added
	TransportEvent	Added
Signature		

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	Note	Changed cardinality from 0..1 to 0..n
	ValidatorID	Changed dictionary entry name from "Signature. Validator Identifier. Identifier" to "Signature. Validator. Identifier"
	SignatoryParty	Changed cardinality from 1 to 0..1
StatementLine		
	Note	Changed cardinality from 0..1 to 0..n
	PaymentPurposeCode	Added
	AllowanceCharge	Added
	CollectedPayment	Added
Status		
	ReferenceDate	Changed dictionary entry name from "Status. Reference_ Date. Date" to "Status. Reference Date. Date"
	ReferenceTime	Changed dictionary entry name from "Status. Reference_ Time. Time" to "Status. Reference Time. Time"
	Description	Changed cardinality from 0..1 to 0..n
	StatusReason	Changed cardinality from 0..1 to 0..n
	SequenceID	Changed dictionary entry name from "Status. Sequence. Identifier" to "Status. Sequence Identifier. Identifier"
	Text	Changed cardinality from 0..1 to 0..n
	ConditionValueMeasure	Added
	ReliabilityPercent	Added
StockAvailabilityReport-Line		Added
SubcontractTerms		Added
SubscriberConsumption		Added
SupplierConsumption		Added
TaxTotal		
	TaxIncludedIndicator	Added
TelecommunicationsService		Added
TelecommunicationsSupply		Added
TelecommunicationsSupplyLine		Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
TenderLine		Added
TenderPreparation		Added
TenderRequirement		Added
TenderResult		Added
TenderedProject		Added
TendererPartyQualification		Added
TendererQualificationRequest		Added
TendererRequirement		Added
TenderingProcess		Added
TenderingTerms		Added
TradeFinancing		Added
TransportEquipment		
	ReferencedConsignmentID	Added
	AirFlowPercent	Added
	HumidityPercent	Added
	AnimalFoodApprovedIndicator	Added
	HumanFoodApprovedIndicator	Added
	DangerousGoodsApprovedIndicator	Added
	RefrigeratedIndicator	Added
	Characteristics	Added
	DamageRemarks	Added
	Description	Added
	SpecialTransportRequirments	Added
	GrossWeightMeasure	Added
	GrossVolumeMeasure	Added
	TareWeightMeasure	Added
	TrackingDeviceCode	Added
	PowerIndicator	Added
	TraceID	Added
	SupplierParty	Added
	OwnerParty	Added
	OperatingParty	Added
	UnloadingLocation	Added
	StorageLocation	Added
	PositioningTransportEvent	Added
	QuarantineTransportEvent	Added
	DeliveryTransportEvent	Added
	PickupTransportEvent	Added
	HandlingTransportEvent	Added
	LoadingTransportEvent	Added
	TransportEvent	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	ApplicableTransportMeans	Added
	HaulageTradingTerms	Added
	HazardousGoodsTransit	Added
	PackagedTransportHandlingUnit	Added
	ServiceAllowanceCharge	Added
	FreightAllowanceCharge	Added
	AttachedTransportEquipment	Added
	GoodsItem	Added
	Delivery	Added
	ContainedInTransportEquipment	Added
	Pickup	Added
	Despatch	Added
	ShipmentDocumentReference	Added
	ContainedPackage	Added
TransportEvent		
	CurrentStatus	Changed cardinality from 1..n to 0..n
	Location	Added
	Signature	Added
	Period	Added
TransportExecutionTerms		Added
TransportHandlingUnit		
	TraceID	Added
	TransportMeans	Added
	GoodsItem	Added
	FloorSpaceMeasurementDimension	Added
	PalletSpaceMeasurementDimension	Added
	ShipmentDocumentReference	Added
	Status	Added
	CustomsIdentification	Added
	ReferencedShipment	Added
	Package	Added
TransportMeans		
	TransportMeansTypeCode	Added
	TradeServiceCode	Added
	DriverParty	Added
	PassengerParty	Added
	ReportingParty	Added
	MeasurementDimension	Added
TransportSchedule		Added
TransportStatus		Added
TransportationSegment		Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
TransportationService		
	TransportationServiceDescription	Added
	TransportationServiceDetailsURI	Added
	NominationDate	Added
	NominationTime	Added
	Name	Added
	SequenceNumeric	Added
	TransportHandlingUnit	Added
	IncludedTransportHandlingUnit	Added
	ExcludedTransportHandlingUnit	Added
	TotalCapacityDimension	Added
	ShipmentStage	Added
	TransportEvent	Added
	ResponsibleTransportServiceProvider-Party	Added
	EnvironmentalEmission	Added
	EstimatedDurationPeriod	Added
UnstructuredPrice		Added
UtilityItem		Added
WebSiteAccess		Added
WorkPhaseReference		Added

B.3.4.2 Changes to Document Elements, UBL 2.0 to UBL 2.1

The following table sums up the differences between the XML elements in the UBL 2.0 document schemas (as updated in 2008) and those in the UBL 2.1 document schemas.

Table B.2. Changes to Document Elements, UBL 2.0 to UBL 2.1

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
ApplicationResponse		
	ProfileExecutionID	Added
	VersionID	Changed dictionary entry name from "Application Response. Version Identifier. Identifier" to "Application Response. Version. Identifier"
	DocumentResponse	Changed cardinality from 1..n to 0..n
AttachedDocument		
	ProfileExecutionID	Added
	ParentDocumentVersionID	Added
	ParentDocumentLineReference	Added
AwardedNotification		Added
BillOfLading		
	ProfileExecutionID	Added
CallForTenders		Added
Catalogue		
	ProfileExecutionID	Added
	ActionCode	Added
	SourceCatalogueReference	Added
	DocumentReference	Added
CatalogueDeletion		
	ProfileExecutionID	Added
	EffectiveDate	Added
	EffectiveTime	Added
CatalogueItemSpecificationUpdate		
	ProfileExecutionID	Added
CataloguePricingUpdate		
	ProfileExecutionID	Added
CatalogueRequest		
	ProfileExecutionID	Added
	Signature	Added
CertificateOfOrigin		
	ProfileExecutionID	Added
	VersionID	Changed dictionary entry name from "Certificate Of Origin. Version Identifier. Identifier" to "Certificate Of Origin. Version. Identifier"
	Signature	Added
ContractAwardNotice		Added
ContractNotice		Added
CreditNote		

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	ProfileExecutionID	Added
	CreditNoteTypeCode	Added
	StatementDocumentReference	Added
	OriginatorDocumentReference	Added
	BuyerCustomerParty	Added
	SellerSupplierParty	Added
	Delivery	Added
	DeliveryTerms	Added
	PaymentMeans	Added
	PaymentTerms	Added
DebitNote		
	ProfileExecutionID	Added
	StatementDocumentReference	Added
	BuyerCustomerParty	Added
	SellerSupplierParty	Added
	AllowanceCharge	Added
	Delivery	Added
	DeliveryTerms	Added
	PaymentMeans	Added
	PaymentTerms	Added
DespatchAdvice		
	ProfileExecutionID	Added
DocumentStatus		Added
DocumentStatusRequest		Added
ExceptionCriteria		Added
ExceptionNotification		Added
Forecast		Added
ForecastRevision		Added
ForwardingInstructions		
	ProfileExecutionID	Added
FreightInvoice		
	ProfileExecutionID	Added
	Shipment	Changed cardinality from 1 to 1..n
FulfilmentCancellation		Added
GoodsItemItinerary		Added
GuaranteeCertificate		Added
InstructionForReturns		Added
InventoryReport		Added
Invoice		
	ProfileExecutionID	Added
	DueDate	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	StatementDocumentReference	Added
	ProjectReference	Added
	WithholdingTaxTotal	Added
ItemInformationRequest		Added
Order		
	ProfileExecutionID	Added
	SalesOrderID	Changed dictionary entry name from "Order. Sales Order Identifier. Identifier" to "Order. Sales_ Order Identifier. Identifier"
	OrderTypeCode	Added
	CustomerReference	Changed dictionary entry name from "Order. Customer Reference. Text" to "Order. Customer_ Reference. Text"
	CatalogueReference	Added
	ProjectReference	Added
	PaymentMeans	Changed cardinality from 0..1 to 0..n
	PaymentTerms	Added
	TaxExchangeRate	Added
	PricingExchangeRate	Added
	PaymentExchangeRate	Added
OrderCancellation		
	ProfileExecutionID	Added
OrderChange		
	ProfileExecutionID	Added
	SalesOrderID	Changed dictionary entry name from "Order Change. Sales Order Identifier. Identifier" to "Order Change. Sales_ Order Identifier. Identifier"
	SequenceNumberID	Changed dictionary entry name from "Order Change. Sequence_ Number. Identifier" to "Order Change. Sequence Number. Identifier"
	CustomerReference	Changed dictionary entry name from "Order Change. Customer Reference. Text" to "Order Change. Customer_ Reference. Text"
	PaymentMeans	Changed cardinality from 0..1 to 0..n
	PaymentTerms	Added
	TaxExchangeRate	Added
	PricingExchangeRate	Added
	PaymentExchangeRate	Added
OrderResponse		
	ProfileExecutionID	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	SalesOrderID	Changed dictionary entry name from "Order Response. Sales Order Identifier. Identifier" to "Order Response. Sales_ Order Identifier. Identifier"
	OrderResponseCode	Added
	CustomerReference	Changed dictionary entry name from "Order Response. Customer Reference. Text" to "Order Response. Customer_ Reference. Text"
	PaymentMeans	Changed cardinality from 0..1 to 0..n
	PaymentTerms	Added
	TaxExchangeRate	Added
	PricingExchangeRate	Added
	PaymentExchangeRate	Added
	OrderLine	Changed cardinality from 1..n to 0..n
OrderResponseSimple		
	ProfileExecutionID	Added
PackingList		
	ProfileExecutionID	Added
	VersionID	Changed dictionary entry name from "Packing List. Version Identifier. Identifier" to "Packing List. Version. Identifier"
PerformanceHistory		Added
PriorInformationNotice		Added
ProductActivity		Added
Quotation		
	ProfileExecutionID	Added
ReceiptAdvice		
	ProfileExecutionID	Added
	ReceiptAdviceTypeCode	Added
Reminder		
	ProfileExecutionID	Added
RemittanceAdvice		
	ProfileExecutionID	Added
RequestForQuotation		
	ProfileExecutionID	Added
	SubmissionDueDate	Added
	RequestedValidityPeriod	Added
	BuyerCustomerParty	Added
RetailEvent		Added
SelfBilledCreditNote		
	ProfileExecutionID	Added
	PaymentCurrencyCode	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	PaymentAlternativeCurrencyCode	Added
	StatementDocumentReference	Added
	OriginatorDocumentReference	Added
	BuyerCustomerParty	Added
	SellerSupplierParty	Added
	Delivery	Added
	DeliveryTerms	Added
	PaymentMeans	Added
	PaymentTerms	Added
	PaymentExchangeRate	Added
	PaymentAlternativeExchangeRate	Added
SelfBilledInvoice		
	ProfileExecutionID	Added
Statement		
	ProfileExecutionID	Added
	StatementTypeCode	Added
	PaymentMeans	Changed cardinality from 0..1 to 0..n
StockAvailabilityReport		Added
Tender		Added
TenderReceipt		Added
TendererQualification		Added
TendererQualificationResponse		Added
TradeItemLocationProfile		Added
TransportExecutionPlan		Added
TransportExecutionPlanRequest		Added
TransportExecutionStatus		Added
TransportProgressStatus		Added
TransportProgressStatusRequest		Added
TransportServiceDescription		Added
TransportServiceDescriptionRequest		Added
TransportationStatus		
	ProfileExecutionID	Added
	TransportationStatusTypeCode	Added
	TransportExecutionStatusCode	Added
	TransportationStatusRequestDocumentReference	Added
	TransportExecutionPlanDocumentReference	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	Consignment	Changed cardinality from 1 to 0..n
	TransportEvent	Changed cardinality from 1..n to 0..n
	UpdatedPickupTransportEvent	Added
	UpdatedDeliveryTransportEvent	Added
	StatusLocation	Added
	StatusPeriod	Added
TransportationStatus-Request		Added
UnawardedNotification		Added
UtilityStatement		Added
Waybill		
	ProfileExecutionID	Added

B.3.4.3 Changes to Attributes, UBL 2.0 to UBL 2.1

The following table lists all the attributes added since the release of UBL 2.0 (as updated in 2008).

Table B.3. Changes to Attributes, UBL 2.0 to UBL 2.1

Type	Attribute	Changes for UBL 2.1
Amount		
	CurrencyCodeListVersionID	Added
Measure		
	UnitCodeListVersionID	Added
Numeric		
	format	Added
Percent		
	format	Added
Rate		
	format	Added
Quantity		
	UnitCodeListID	Added
	UnitCodeListAgencyID	Added
	UnitCodeListAgencyName	Added
Text		
	languageLocalID	Added
Name		
	languageLocalID	Added

B.3.5 Changes from UBL 2.1 PRD2 XML to Final UBL 2.1 XML

This section documents changes made in the final round of UBL 2.1 development prior to release, that is, changes made following the Second Public Review Draft (PRD2) of this specification. The material herein is of relevance only to test implementers of the former draft, who should **note that changes to the cardinality of certain attributes make UBL 2.1 as released technically backward incompatible with PRD2, though it remains fully backward-compatible with UBL 2.0.** See Section B.3.5.3, “Changes to Attributes Following PRD2”. This section is otherwise of only historical interest.

B.3.5.1 Changes to Library Elements Following PRD2

The following table sums up the differences between the XML elements in the Common Library of the Second Review Draft and the XML elements in the UBL 2.1 Common Library of this distribution.

Table B.4. Changes to Library Elements Following PRD2

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
CatalogueLine		
	ReplacedRelatedItem	Added
Clause		Added
Consignment		
	CustomsIdentification	Added
	RequestedPickupTransportEvent	Added
	RequestedDeliveryTransportEvent	Added
	PlannedPickupTransportEvent	Added
	PlannedDeliveryTransportEvent	Added
	Status	Added
	ChildConsignment	Added
	MainCarriageShipmentStage	Changed cardinality from 0..1 to 0..n
ConsumptionPoint		
	SubscriberType	Added
CustomsIdentification		Added
Duty		
	Duty	Changed dictionary entry name from "Duty. Details Duty. Duty. Text" to "Duty. Details"
	Duty	Changed dictionary entry name from "Duty. Details Duty. Duty. Text" to "Duty. Duty. Text"
EmissionCalculation-Method		Added
EnvironmentalEmission		
	EmissionCalculationMethod	Added
GoodsItem		
	TraceID	Added
InvoiceLine		
	WithholdingTaxTotal	Added
ItemIdentification		
	BarcodeSymbologyID	Added
Location		
	Description	Changed cardinality from 0..1 to 0..n
	Name	Added
MaritimeTransport		
	RadioCallSignCode	Added
MeterReading		
	ID	Added
	PreviousMeterReadingMethod	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	PreviousMeterReadingMethodCode	Added
	LatestMeterReadingMethod	Added
	LatestMeterReadingMethodCode	Added
OrderLine		
	OrderLineReference	Added
PaymentMandate		
	Clause	Changed dictionary entry name from "Payment Mandate. Clause. Text" to "Payment Mandate. Clause"
Person		
	OtherName	Added
	BirthplaceName	Added
	ResidenceAddress	Added
ProjectReference		Added
Shipment		
	Consignment	Changed cardinality from 1..n to 0..n
ShipmentStage		
	PreCarriageIndicator	Changed cardinality from 1 to 0..1
	OnCarriageIndicator	Changed cardinality from 1 to 0..1
	EstimatedTransitPeriod	Added
	FreightAllowanceCharge	Added
	FreightChargeLocation	Added
	DetentionTransportEvent	Added
	RequestedDepartureTransportEvent	Added
	RequestedArrivalTransportEvent	Added
	RequestedWaypointTransportEvent	Added
	PlannedDepartureTransportEvent	Added
	PlannedArrivalTransportEvent	Added
	PlannedWaypointTransportEvent	Added
	TransportEvent	Added
StatementLine		
	CollectedPayment	Added
SupplierConsumption		
	Contract	Added
TradeFinancing		
	Clause	Changed dictionary entry name from "Trade Financing. Clause. Text" to "Trade Financing. Clause"
TransportEquipment		
	ContainedPackage	Added
TransportEvent		

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	CurrentStatus	Changed cardinality from 1..n to 0..n
	Signature	Added
	Period	Added
TransportExecu-tionTerms		
	PaymentTerms	Changed cardinality from 1 to 0..n
	ServiceChargePaymentTerms	Added
TransportHandlingUnit		
	TraceID	Added
	CustomsIdentification	Added
	ReferencedShipment	Added
	Package	Added
TransportMeans		
	DriverParty	Added
	PassengerParty	Added
	ReportingParty	Added
	MeasurementDimension	Added
TransportSchedule		
	SequenceNumeric	Added
	ReferenceDate	Changed cardinality from 1 to 0..1
	ReferenceTime	Changed cardinality from 1 to 0..1
	EstimatedDepartureTransportEvent	Added
	EstimatedArrivalTransportEvent	Added
	PlannedDepartureTransportEvent	Added
	PlannedArrivalTransportEvent	Added
TransportationSegment		
	SequenceNumeric	Added
	TransportExecutionPlanReferenceID	Changed cardinality from 1 to 0..1
	ReferencedConsignment	Added
	ShipmentStage	Added
TransportationService		
	Name	Added
	SequenceNumeric	Added
	TransportHandlingUnit	Added
	IncludedTransportHandlingUnit	Added
	ExcludedTransportHandlingUnit	Added
	TotalCapacityDimension	Added
	ShipmentStage	Added
	TransportEvent	Added
	ResponsibleTransportServiceProvider-Party	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	EnvironmentalEmission	Added
	EstimatedDurationPeriod	Added
WorkPhaseReference		Added

B.3.5.2 Changes to Document Elements Following PRD2

The following table sums up the differences between the XML elements in the UBL 2.1 document schemas of this distribution and the XML elements in the document schemas of the Second Review Draft.

Table B.5. Changes to Document Elements Following PRD2

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
CreditNote		
	CreditNoteTypeCode	Added
DocumentStatus		
	DocumentResponse	Changed cardinality from 1 to 0..1
FreightInvoice		
	Shipment	Changed cardinality from 1 to 1..n
FulfilmentCancellation		Added
GoodsItemItinerary		
	ReferencedConsignment	Added
Invoice		
	ProjectReference	Added
	WithholdingTaxTotal	Added
Order		
	ProjectReference	Added
OrderResponse		
	OrderLine	Changed cardinality from 1..n to 0..n
ReceiptAdvice		
	ReceiptAdviceTypeCode	Added
TransportExecutionPlan		
	VersionID	Added
	DocumentStatusCode	Added
	DocumentStatusReasonCode	Added
	DocumentStatusReasonDescription	Added
	SenderParty	Added
	ReceiverParty	Added
	TransportExecutionPlanRequestDocumentReference	Added
	TransportExecutionPlanDocumentReference	Added
	Consignment	Added
TransportExecutionPlanRequest		Added
TransportProgressStatus		
	TransportProgressStatusRequestDocumentReference	Added
TransportProgressStatusRequest		Added
TransportServiceDescription		
	TransportServiceDescriptionRequestDocumentReference	Added

Aggregate BIE	Basic or Association BIE	Changes for UBL 2.1
	ServiceChargePaymentTerms	Added
	TransportationService	Added
TransportServiceDescription-Request		Added
TransportationStatus		
	TransportationStatusTypeCode	Added
	TransportExecutionStatusCode	Added
	TransportationStatusRequestDocu-mentReference	Added
	TransportExecutionPlanDocumentRefer-ence	Added
	Consignment	Changed cardinality from 1 to 0..n
	TransportEvent	Changed cardinality from 1..n to 0..n
	UpdatedPickupTransportEvent	Added
	UpdatedDeliveryTransportEvent	Added
	StatusLocation	Added
	StatusPeriod	Added
TransportationStatus-Request		Added

B.3.5.3 Changes to Attributes Following PRD2

The following attributes, which were optional in UBL 2.1 PRD2, have been changed from optional to mandatory in the this release.

The `currencyID` attribute on `AmountType`
The `mimeCode` attribute on `BinaryObjectType`
The `unitCode` attribute on `MeasureType`

These changes bring UBL 2.1 back into alignment with UBL 2.0, in which these attributes were also mandatory. **As a consequence, however, UBL 2.1 is not backward-compatible with UBL 2.1 PRD2 in this respect.**

B.3.5.4 Other Changes Following PRD2

The context/value association file `cva/UBL-DefaultDTQ-2.1.cva` has been revised to repair an absent check for the schemeID attribute for identifiers.

Appendix C The UBL 2.1 Data Model (Non-Normative)

Following the principles of the ebXML Core Components Technical Specification [**CCTS**], the UBL data model is based on a library of reusable information items known as Business Information Entities (BIEs). Each business document defined by UBL is created by assembling items appropriate to that document type from the UBL BIE library. Further detail regarding BIEs is provided in Section C.3, "Business Information Entities".

C.1 UBL 2.1 Schema Generation

Historically, both the UBL common library of reusable components and the assembly models for the individual UBL documents have been expressed as spreadsheets using a format specifically developed for UBL business information modeling (this format is discussed further below). Each spreadsheet is provided in both Open Document and Microsoft Excel formats; in the UBL 2.1 release, the spreadsheets for the common library will be found in `mod/common`, and spreadsheets for the individual documents will be found in `mod/maindoc`.

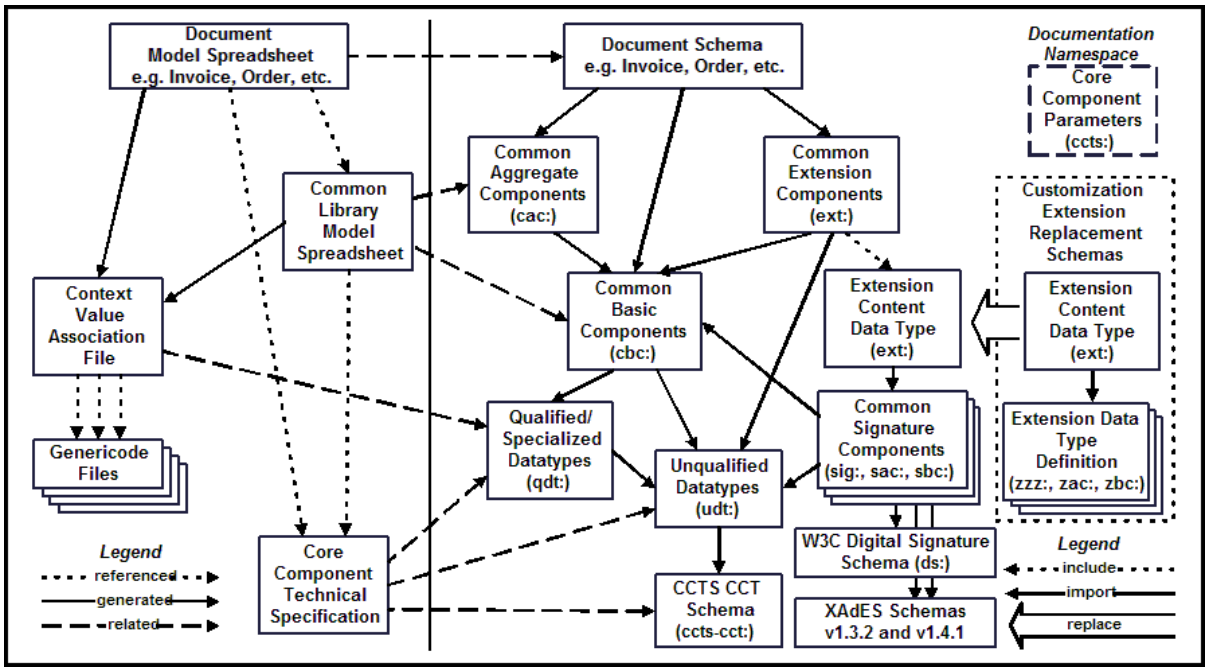
In early releases of UBL, the XSD schemas that serve as the normative expression of UBL syntax were generated directly from spreadsheets handcrafted by business experts using a spreadsheet-to-schema transformation model specified by the UBL Naming and Design Rules, which are typically released as a separate OASIS Committee Specification some time after each version of UBL itself. In UBL 2.0, the entire data model was also entered (via the spreadsheets) into the internal format of a commercial data management system from GEFEG [http://gefeg.com/] that was used to insure data integrity.

In UBL 2.1, the data model is instantiated and maintained in iSurf eDoCreator [http://ubl.xml.org/blog/isurf-edocreator-e-business-document-design-and-customization-environment], a project of the European Commission, and spreadsheets using the format that has become standard for UBL development are generated from the eDoCreator database. The UBL 2.1 schemas are, in turn, generated from these spreadsheets using a set of XSLT scripts created and maintained by the UBL Schema Generation Task Group. Q/A for these schemas is aided by a comparison with schemas independently generated by eDoCreator from the same database used to generate the spreadsheets.

This methodology continues the role of the UBL spreadsheets in providing base-level human-readable documentation of the UBL data model and publishing the supplementary metadata required by [**CCTS**]. By preserving a vendor-neutral representation from which schemas can be generated directly, the spreadsheets guarantee that the UBL model is not bound to a single production system.

The following diagram shows the conceptual relationships between spreadsheets and schemas in UBL 2.1, with spreadsheets on the left and schemas on the right. Compare Figure 63, "UBL Schema Dependencies".

Figure C.1. UBL Spreadsheet Realization



C.2 The UBL Common Library

As noted above, UBL is based on a reusable library of Business Information Entities. In the current release, the Common Library contains more than two thousand of these individually defined data items. The entire UBL 2.1 library of Business Information Entities is contained in a single spreadsheet, provided here like all the spreadsheets in two different formats.

```
mod/common/UBL-CommonLibrary-2.1.ods
mod/common/UBL-CommonLibrary-2.1.xls
```

C.3 Business Information Entities

In the language of [**CCTS**], UBL Business Information Items (BIEs) include BBIEs (“basic” individual pieces of information), ABIEs (aggregations of other BIEs), and ASBIEs (“associations” to ABIEs). Fuller explanations of these terms in the context of the CCTS framework will be found in the CCTS specification. For purposes of understanding UBL as a set of XML schemas, however, it may be useful to describe these terms employing concepts more familiar to XML users.

With the understanding that every XML document describes a logical tree of elements, the different kinds of Business Information Entities from which UBL documents are constructed may be described as follows:

UBL BBIEs (Basic Business Information Entities) are the leaf nodes of every UBL document structure. These are ordinary data fields such as one would expect to find in any business form, and they are realized in the schemas as individual XML elements at the bottom level of the document tree with simple content representing amounts, codes, quantities, and so on. All UBL BBIE elements (and only UBL BBIE elements) are members of the UBL common basic components namespace, conventionally denoted in UBL schemas by the `cbc:` prefix. (Since all namespace prefixes in XML are assigned on a per-instance basis according to namespace declarations in the individual instance, prefixes such as `cbc:` may be replaced with arbitrarily different namespace prefixes in actual UBL documents.)

UBL ASBIEs (Association Business Information Entities) are substructures of a UBL document. Children of ASBIEs may be BBIEs or other ASBIEs, never ABIEs. All UBL ASBIEs (and only UBL ASBIEs)

are members *as elements* of the UBL common aggregate components namespace, denoted in UBL schemas by the `cac:` prefix.

UBL document ABIEs (Aggregate Business Information Entities) are the root nodes and top-level structures of UBL documents. Children of document ABIEs may be BBIEs or ASBIEs, never ABIEs. All UBL document ABIEs (and only UBL document ABIEs) are defined within individual namespaces specific to each document as both elements and types.

UBL library ABIEs (that is, all ABIEs except document ABIEs) have a structural shape but are not concrete document structures; rather, they are abstract structures or templates for ASBIEs, thus allowing the same structure to be reused in multiple roles. Children of library ABIEs in the data structure can be BBIEs or ASBIEs, never ABIEs. All library ABIEs must be realized as ASBIEs in order to actually exist as elements in the UBL document tree. All UBL library ABIEs (and only UBL library ABIEs) are realized *as types* in the UBL `cac:` namespace.

This naming scheme inherited from CCTS may prove problematic for some UBL users. In particular, the CCTS terms “Association Business Information Entity” and “Aggregate Business Information Entity” do not well describe these two concepts as they are realized in XML. The problem word here is “association”, which correctly describes this relationship within a UML (Unified Modeling Language) framework but is perhaps better thought of in the UBL context as meaning that a particular ASBIE is “associated with” an abstract ABIE structure. For our purposes, it would have been better if ASBIEs had instead been called “Aggregate Business Information Entities” and ABIEs had instead been called “Structural Templates”. It may prove easiest for the UBL user to regard the terms ASBIE and ABIE as opaque labels and to ignore the historical expansions of these acronyms.

It can be seen from the above that the XML implementations of ASBIEs and library ABIEs share the same `cac:` namespace. In the schemas, library ABIEs are all implemented as XML types, and ASBIEs are all implemented as XML elements. This is simply a reflection of their different roles—library ABIEs as abstract classes or structural templates (realized as XML types) and ASBIEs as concrete instantiations (realized as XML elements derived from those types).

While the distinction between ABIEs/classes/types on the one hand and ASBIEs/instantiations/elements on the other is clear enough, it should be noted that in some cases an ASBIE does not qualify the name of the ABIE from which it is derived. In effect, they have the same name. Some library ABIEs are used only in the form of an ASBIE having the same name; for example, `AddressLine` is a library ABIE that is only used in the form of an ASBIE named `AddressLine`. Some library ABIEs are realized in some places as ASBIEs with the same name (where it is felt that the unqualified name is sufficient) and elsewhere as ASBIEs with a name that is further qualified; for example, the library ABIE `Address` has numerous ASBIE realizations with qualified names like `LocationAddress`, `ApplicableAddress`, `DespatchAddress`, and so on, but it's also seen as an ASBIE simply named `Address` that's included in the library ABIEs `FinancialInstitution`, `Branch`, `Location`, and `ConsumptionPoint`. Some library ABIEs are never actually implemented as ASBIEs with the same name; for example, only one ASBIE is associated with the library ABIE `ActivityDataLine`, and it has the qualified name `SupplyChainActivityDataLine`.

The UBL Common Aggregate Component schema declares an identically named element or potential ASBIE for every library ABIE regardless of whether that element is used in a UBL document schema to represent an ASBIE (these are among the long list of global element declarations at the beginning of the CAC module). ABIEs are implemented as one or more ASBIEs via XSD references to these elements farther down in the CAC schema module or in individual document schema modules, which all import the CAC module. For example, the global element `AddressLine` declared in the CAC with the line

```
<xsd:element name="AddressLine" type="AddressLineType"/>
```

is implemented as an ASBIE with the same name in the declaration of the `Address` ABIE as follows:

```
<xsd:element ref="cac:AddressLine" minOccurs="0" maxOccurs="unbounded">
   [...]
</xsd:element>
```

One consequence of this approach is that the list of global elements that begins the CAC module contains elements that are in fact never used under those names in UBL 2.1. For example, the element `ActivityDataLine` mentioned above is used by reference in creating the ASBIE `SupplyChainActivityDataLine`, but it never appears in the form of an ASBIE named `ActivityDataLine`. Such unused ABIE names remain available in the global element declarations for customizers and designers of future additions to UBL.

C.4 Navigating the UBL 2.1 Model Spreadsheets

The concepts described above can be illustrated by navigating the UBL data models to construct a trivial UBL Invoice instance.

We will start with a wrapper copied from an example in the `xml` directory of the UBL 2.1 distribution (`xml/UBL-Invoice-2.1-Example.xml`):

```
<?xml version="1.0" encoding="UTF-8"?>
<Invoice xmlns="urn:oasis:names:specification:ubl:schema:xsd:Invoice-2"
xmlns:cac="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2"
xmlns:cbc="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2">
[...]
</Invoice>
```

Now we will fill out this shell of an instance, completing the part in the brackets by traversing the data model.

Like all the document models defined by UBL, the Invoice model is expressed as a spreadsheet. UBL provides both .ods and .xls versions:

`mod/maindoc/UBL-Invoice-2.1.ods`
`mod/maindoc/UBL-Invoice-2.1.xls`

Note that the spreadsheets corresponding to each UBL 2.1 document type are linked from the description of that type in Section 3.1, “UBL 2.1 Document Schemas”.

Line 2 of the spreadsheet (either version) defines the document ABIE named `Invoice`. Column P confirms that `Invoice` is an ABIE, as also indicated by the pink background in that row of the spreadsheet.

Everything after `Invoice` in the spreadsheet ends up as part of the schema, and the order seen here is the order in which these components will appear in both the schema and any conforming instances of `Invoice`. The BBIE children of `Invoice` are given first (white background), and then all the ASBIE children of `Invoice` (green background).

As shown in Column O, most of these components are optional. The first required field is `ID` and the second is `IssueDate`, so we can write, for example,

```
  <cbc:ID>123</cbc:ID>
  <cbc:IssueDate>2011-09-22</cbc:IssueDate>
```

Next let's add an optional `InvoicePeriod`. This is an ASBIE, implying that it has some kind of substructure, and it derives from the generic ABIE called `Period` (this is the “Associated Object Class” referred to in Column M). To find this structure, we look for the `Period` ABIE in one of the Common Library model spreadsheets

`mod/common/UBL-CommonLibrary-2.1.ods` or
`mod/common/UBL-CommonLibrary-2.1.xls`

`Period` will be found at line 1352 and seen to contain seven possible BBIE children, all of them optional; and the ASBIE `InvoicePeriod` in `Invoice` therefore has this structure, too. From this one could

conclude that instantiations of the Period structure (there are more than 50 of them in UBL 2.1) need not contain any of the seven optional BBIE elements specified at lines 1353–1359, and indeed the corresponding declaration of the complex type PeriodType in the CAC schema (xsd/common/UBL-CommonAggregateComponents-2.1.xsd) shows that an empty InvoicePeriod element will pass XML validation; but UBL explicitly prohibits such structures (see Section 4.3, "Empty Elements"). In UBL, as a normative rule independent of schema constraints, every ASBIE must have at least one child (BBIE or ASBIE) instantiated. In this case, therefore, one or more of the seven possible BBIE children of InvoicePeriod will need to appear in a UBL Invoice document for it to be conformant UBL in addition to the requirement that the document validate against the Invoice schema. If StartDate and EndDate (for example) are chosen for the content of InvoicePeriod, the corresponding section of the sample instance might then look like this:

```
<cac:InvoicePeriod>
  <cbc:StartDate>2011-08-01</cbc:StartDate>
  <cbc:EndDate>2011-08-31</cbc:EndDate>
</cac:InvoicePeriod>
```

Next in order come two required pieces, the ASBIEs AccountingSupplierParty and AccountingCustomerParty. As shown in Column M of the Invoice spreadsheet, AccountingSupplierParty (line 34) derives from the SupplierParty ABIE and AccountingCustomerParty (line 35) derives from the CustomerParty ABIE. Checking in the Common Library, it is seen that both SupplierParty (line 1784 of the Common Library spreadsheet) and CustomerParty (line 526 of the Common Library spreadsheet) can contain an ASBIE named Party (as shown in lines 1788 and 530, respectively) and that each Party ASBIE is an instantiation of the Party ABIE (line 1246). Therefore both parties have the same structure (lines 1247 through 1265). Thus AccountingSupplierParty and AccountingCustomerParty share the information components common to parties in general and differ in the information peculiar to suppliers and customers. Parties commonly have a PartyName (line 1254) that derives (Column M) from the ABIE PartyName (line 1282), which is a wrapper for the BBIE Name (line 1283). A conforming piece of the document instance might therefore look like this:

```
<cac:AccountingSupplierParty>
  <cac:Party>
    <cac:PartyName>
      <cbc:Name>Custom Cotter Pins</cbc:Name>
    </cac:PartyName>
  </cac:Party>
</cac:AccountingSupplierParty>
<cac:AccountingCustomerParty>
  <cac:Party>
    <cac:PartyName>
      <cbc:Name>North American Veeblefetzer</cbc:Name>
    </cac:PartyName>
  </cac:Party>
</cac:AccountingCustomerParty>
```

Returning to the Invoice model, it is seen that the Invoice must close with a LegalMonetaryTotal (line 51) and at least one InvoiceLine (line 52). Taking LegalMonetaryTotal first, it is found in the Common Library to be derived from MonetaryTotal (line 1182), which has a mandatory PayableAmount BBIE. A corresponding example instance fragment might be therefore be constructed as follows:

```
<cac:LegalMonetaryTotal>
   <cbc:PayableAmount currencyID="CAD">100.00</cbc:PayableAmount>
</cac:LegalMonetaryTotal>
```

If the preceding explanation of Party is understood, there should be nothing problematic about the process of forming the example LegalMonetaryTotal element except the currencyID attribute on

PayableAmount, which does not appear explicitly in the model line for that BBIE (line 1190). This is because UBL does not define the primitive data types upon which the model is built; instead it uses standard data type definitions from [**CCTS**] and [**XSD2**]. In the case of PayableAmount, the CCTS data type (Column K) is "Amount. Type" (the space is part of the name), and that type is defined in [**CCTS**] itself (Table 8-1 of the CCTS 2.01 specification). There it will be seen that "Amount. Type" has two supplementary "CCT Components" called "Amount. Currency. Identifier" and "Currency. Code List Version. Identifier". In the XML realization of CCTS, supplementary components are expressed as attributes, and the CCTS names "Amount. Currency. Identifier"and "Currency. Code List Version. Identifier" are transformed into the XML attribute names currencyID and currencyCodeListVersionID, respectively. All of these CCTS-based types and attributes are declared in the CCTS Core Component Type schema module:

xsd/common/CCTS_CCT_SchemaModule-2.1.xsd

Note that this schema module comes from UN/CEFACT, not UBL; that it does not implement all of the supplementary components of core component types defined by [**CCTS**]; and that all of the attributes it does declare are defined as optional. In UBL, however, the attributes currencyID and mimeCode are required, not optional. In order to impose its own restrictions, therefore, and also to supply a full set of supplementary component attributes, UBL provides an Unqualified Data Types module that imports the CCTS module and then overrides those definitions as needed:

xsd/common/UBL-UnqualifiedDataTypes-2.1.xsd

Further information about UBL data types can be found in Appendix E, *Data Type Qualifications in UBL (Non-Normative)*. Note in particular Table E.2, "UBL Unqualified Data Types", which includes a list of all the attributes associated with UBL unqualified data types. A reverse lookup of the implied occurrence of each attribute in the data models is provided in this summary report:

mod/summary/reports/UBL-AllDocuments-2.1.html#UDT

In the example fragment above, currencyID has been used to label the amount in Canadian dollars (CAD). As explained in Appendix F, *UBL 2.1 Code Lists and Two-phase Validation (Non-Normative)*, the value CAD for this attribute is not specified in schemas to be checked using XSD validation but will instead be found in separate OASIS genericode code list files in the gc directory of the UBL 2.1 distribution, which are engaged through a separate XSLT-based process.

Using the same methodology, a sample InvoiceLine can be constructed to complete the example as follows:

```
<cac:InvoiceLine>
  <cbc:ID>1</cbc:ID>
  <cbc:LineExtensionAmount currencyID="CAD">100.00</cbc:LineExtensionAmount>
  <cac:Item>
     <cbc:Description>Cotter pin, MIL-SPEC</cbc:Description>
  </cac:Item>
</cac:InvoiceLine>
```

The finished example can be found in

xml/UBL-Invoice-2.1-Example-Trivial.xml

C.5 Summary Reports

While the document model spreadsheets described above function as a basic form of documentation, an alternative form of documentation that will be found useful in many contexts is provided by HTML reports contributed by Crane Softwrights and included here by permission in the mod/summary/reports directory. The summary report for each UBL 2.1 document type is linked from the description of that type in Section 3.1, "UBL 2.1 Document Schemas".

Each document report summarizes business object definitions and selected columns of the corresponding spreadsheet in a hyperlinked form that omits unused elements to facilitate rapid review of each document model. There is also a single master report incorporating every document type and the entire common library:

mod/summary/reports/UBL-AllDocuments-2.1.html

For notes on the use of these reports, and links to abbreviated reports for each of the individual UBL document types, see

mod/summary/readme-Reports.html

C.6 Business Information Entity Documentation

The mod directory also contains a complete list of all the UBL 2.1 business information entities (BBIEs, ABIEs, and ASBIEs) in genericode format and an HTML file displaying information about the re-use of ABIEs and ASBIEs in table form.

mod/UBL-Entities-2.1.gc
mod/UBL-Signature-Entities-2.1.gc
mod/UBL-ABIE-Reuse-Table-2.1.html

Appendix D Notes on Terminology (Non-Normative)

This appendix explains some terms that may be confusing to data technicians unfamiliar with business content.

D.1 Item vs. Line Item

Many of the UBL document types employ the concept of a “line” inherited from traditional paper documents such as purchase orders and invoices. As in these older realizations, a “line” is a substantial data object with a number of subfields, typically including a short description, quantity, unit name, unit price, extension, and so on. Often in UBL these data structures include an element named `Item` that describes more fully the item of sale being ordered, invoiced, shipped, etc. `Item` in the line context always refers to the generic item of sale, not a unique, trackable, individual instance of such an item.

In the case of line structures such as `InvoiceLine` and `TenderLine`, the relationship between the line and the `Item` it contains is unproblematic, but a person unfamiliar with traditional usage may easily be confused by the line element called `LineItem`. In traditional business processes, “line item” is a common name for the entire line structure in a purchase order or invoice, *not just the item of sale contained in the line.* Thus, despite the name, a `LineItem` is not an `Item` but rather a complex data structure that *contains* an `Item` along with quantity, price, and so on.

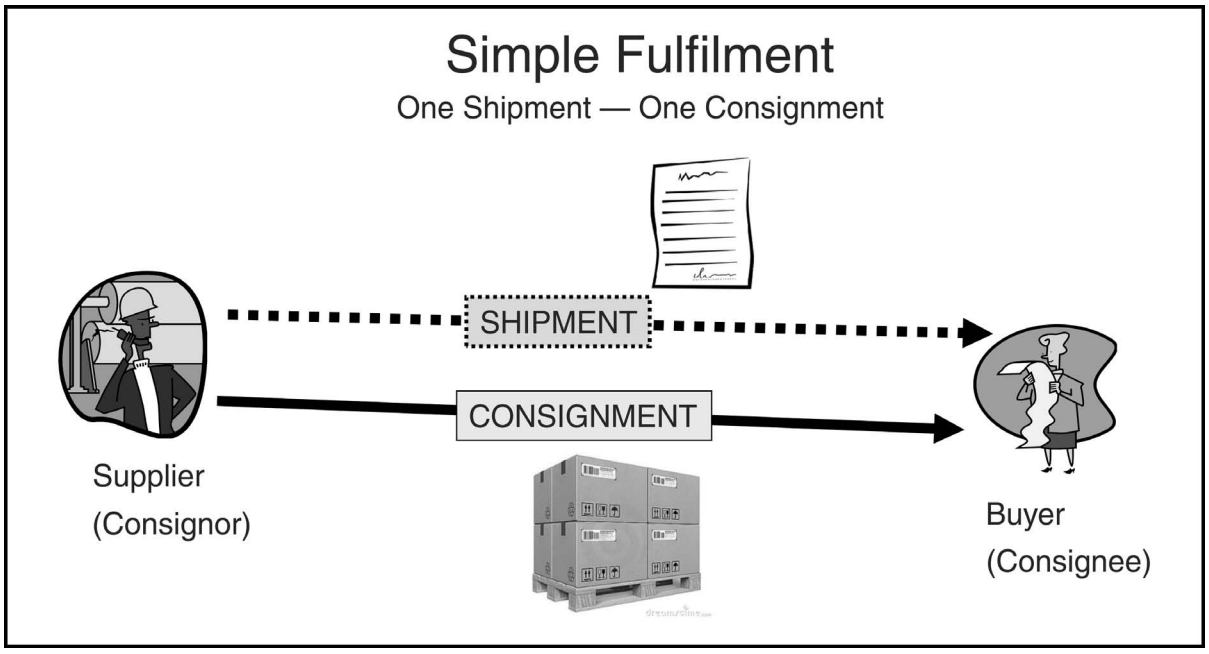
D.2 Shipment vs. Consignment

References to “shipment” and “consignment” appear in a number of places in the UBL data model relating to the transport of goods. For IT specialists unfamiliar with the way these terms are used in international trade, the structural relationships between the two can be puzzling. For example, a close look at the data model shows that shipments can comprise multiple consignments and consignments can comprise multiple shipments. This is not a design flaw but rather a reflection of the possible real-world relationships between the two concepts.

Shipment and *consignment* actually refer to two different ways of looking at the same (possibly very complex) situation. From the physical or logistical point of view, a *consignment* is the transportation of an identifiable collection of goods items from one party (the consignor) to another (the consignee) via one or more modes of transport. From the contractual or logical point of view, a *shipment* is the arrangement whereby an identifiable collection of goods items is to be transported from one party (the shipper) to another party (the recipient). In UBL, the party originating the shipment is usually a supplier, and the party receiving the shipment is usually a buyer.

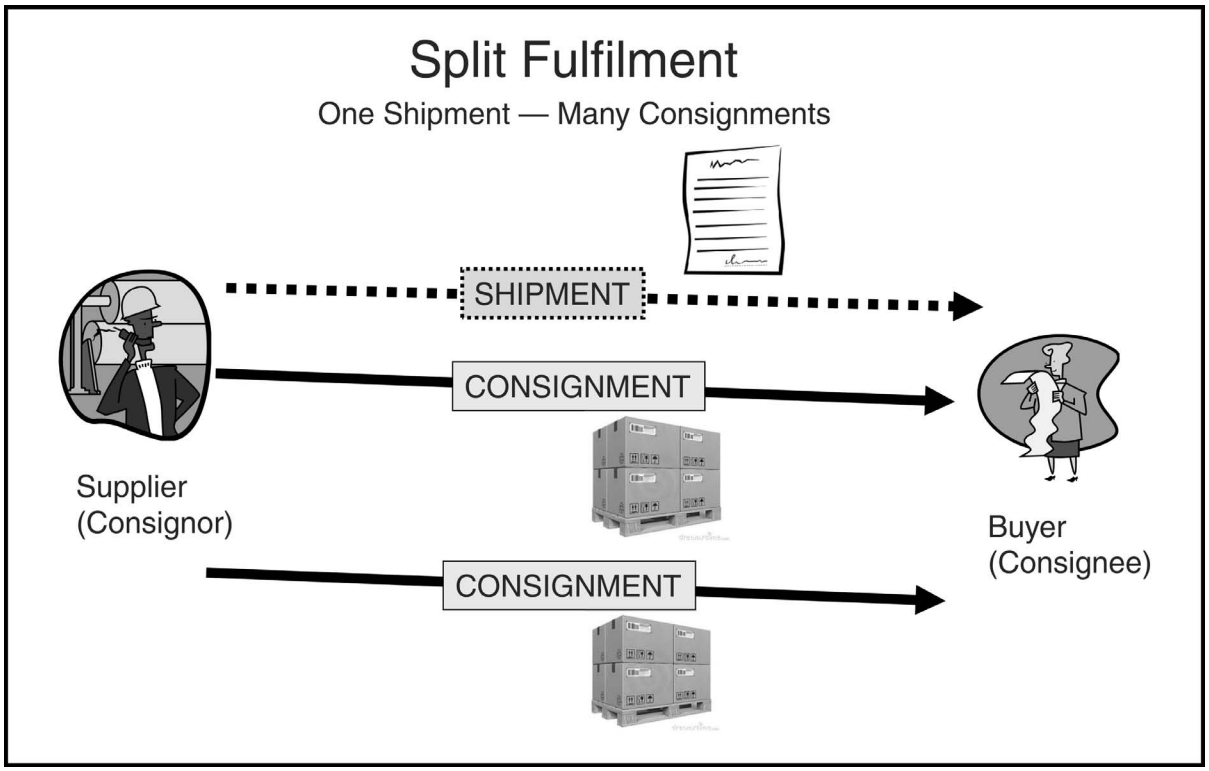
In the simplest fulfilment scenario, these distinctions are almost invisible; see Figure D.1, “Simple Fulfilment” below (used, like the subsequent three, by permission of Document Engineering Services). In this case, the supplier of the contracted shipment is the consignor of the physical goods, and the buyer is the consignee.

Figure D.1. Simple Fulfilment



Often, however, a single contractual shipment is split up into separate physical consignments that may be received on separate schedules, as shown in Figure D.2, “Split Fulfilment”. The shipper may use multiple carriers, or the shipment may be so large that it must be transported in multiple vessels, becoming in effect multiple consignments. It is therefore often necessary for the UBL description of a shipment to contain descriptions of the consignments into which the goods have been divided.

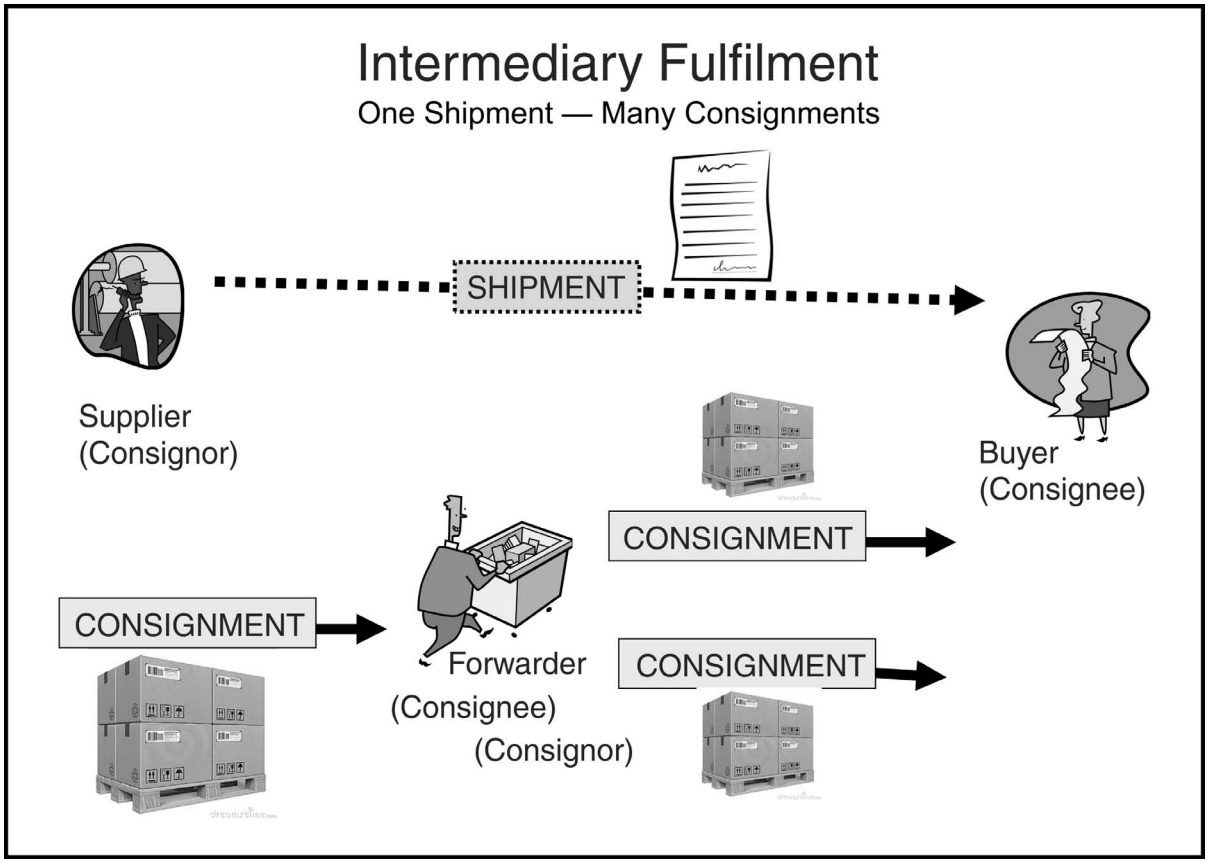
Figure D.2. Split Fulfilment



So far, the shipper (here a supplier) remains the only consignor and the recipient (here the buyer) the only consignee. But sometimes the division of a shipment into consigments takes place “behind the scenes” through the involvement of a freight forwarder, who becomes both a second consignee and a

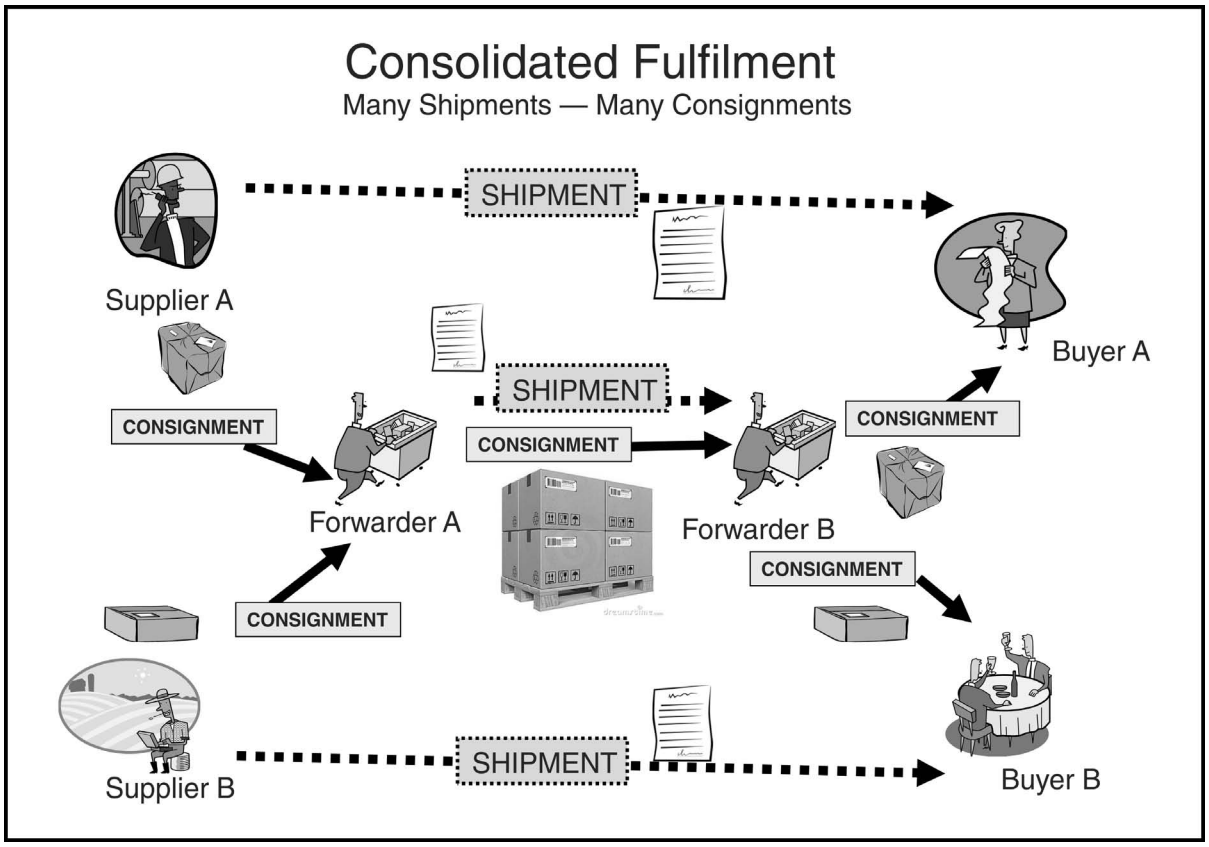
second consignor (Figure D.3, “Intermediary Fulfilment”). The “shipment” in this case is the entire end-to-end organization of the transport of goods on behalf of the shipper.

Figure D.3. Intermediary Fulfilment



Another layer of complexity is introduced when pieces of different, possibly unrelated shipments are consolidated into a single consignment to make the physical process more efficient (to share space in the same shipping container, for example, which optimizes transport by ensuring that the container is fully loaded and also provides a more competitive tariff). In Figure D.4, “Consolidated Fulfilment”, goods from two completely unrelated business transactions between two buyers and their suppliers — two different shipments — are consolidated by a freight forwarder into a single consignment for part of their journey and then separated again by another freight forwarder farther on. This requires the UBL description of the consignment to contain descriptions of the shipments participating in the consolidation. Note that the transaction between the two freight forwarders is itself a shipment (a *consolidated shipment*), and its data structure must be able to describe the two shipments it is covering (Supplier A to Buyer A and Supplier B to Buyer B) so that the receiving forwarder knows how to deconsolidate the consignment.

Figure D.4. Consolidated Fulfilment



Note that the word “consignment” in the context of transportation has a meaning different from that of “consignment” in sales and vendor-managed inventory (Section 2.14, “Vendor Managed Inventory”).

D.3 Transport vs. Transportation

The terms “transport” and “transportation” both appear many times in the UBL data model. There is no semantic difference between these terms as used in UBL; in the context of freight management, they mean exactly the same thing: the conveyance of goods or persons.

“Transportation” is the oldest of the two forms, the noun “transportation” first appearing in written English about 70 years earlier than the noun “transport”. UBL 2.0 adopted “transportation” as the preferred form in terms such as “transportation service” and “transportation status”, but in the process of developing UBL 2.1, which features greatly expanded data representation capabilities for multimodal freight management, it became clear that “transport” is the form to be preferred, both because it is shorter and because it is the more commonly used of the two in international contexts. The decision to adopt “transport” for new usages while preserving backward compatibility with UBL 2.0 by retaining “transportation” in data items from the earlier release has resulted in the mixed terminology seen here.

Appendix E Data Type Qualifications in UBL (Non-Normative)

All UBL data types ultimately derive either from the UN/CEFACT Core Components Technical Specification [**CCTS**] Core Component Types (CCT) or from the W3C Schema specification [**XSD2**] itself; this derivation takes place in the UBL UDT module. The following table lists the CCTS 2.01 Core Component Types.

Table E.1. CCTS Unqualified Data Types

CCTS Data Type	Definition
Amount. Type	A number of monetary units specified in a currency where the unit of currency is explicit or implied.
Binary Object. Type	A set of finite-length sequences of binary octets.
Code. Type	A character string (letters, figures or symbols) that for brevity and/or language independence may be used to represent or replace a definitive value or text of an Attribute together with relevant supplementary information.
Date Time. Type	A particular point in the progression of time together with relevant supplementary information.
Identifier. Type	A character string to identify and distinguish uniquely, one instance of an object in an identification scheme from all other objects in the same scheme together with relevant supplementary information.
Indicator. Type	A list of two mutually exclusive Boolean values that express the only possible states of a Property.
Measure. Type	A numeric value determined by measuring an object along with the specified unit of measure.
Numeric. Type	Numeric information that is assigned or is determined by calculation, counting, or sequencing. It does not require a unit of quantity or unit of measure.
Quantity. Type	A counted number of non-monetary units possibly including fractions.
Text. Type	A character string (i.e. a finite set of characters) generally in the form of words of a language.

The UBL unqualified data types include the CCTS unqualified data types (named according to the UBL Naming and Design Rules) and a few others, as listed in the following table. Some of these (GraphicType, PictureType, SoundType, VideoType, and ValueType) are defined for completeness but not actually used in UBL 2.1.

The rightmost column of this table lists the UBL XML attributes that implement the CCTS supplementary components associated with each CCTS data type. It is important to be aware of these attributes, because they do not appear directly in the UBL data models but are logically implied by data type inheritance and do appear in the UBL XML schemas in accordance with the UBL Naming and Design Rules. As indicated here, a few of the most significant of these supplementary CCTS components become required XML attributes in UBL and will be required in any instance of an element derived from the corresponding type. See Section C.4, “Navigating the UBL 2.1 Model Spreadsheets” for an example of UBL attributes and a further discussion of this point. A reverse lookup of the implied occurrence of each attribute in the data models is provided in this summary report:

`mod/summary/reports/UBL-AllDocuments-2.1.html#UDT`

Table E.2. UBL Unqualified Data Types

UBL Unqualified Data Type	Definition	Attributes
AmountType	A number of monetary units specified using a given unit of currency.	**currencyID (required)** currencyCodeListVersionID
BinaryObjectType	A set of finite-length sequences of binary octets.	format **mimeCode (required)** encodingCode characterSetCode uri filename
GraphicType	A diagram, graph, mathematical curve, or similar representation.	*not used in UBL 2.1*
PictureType	A diagram, graph, mathematical curve, or similar representation.	*not used in UBL 2.1*
SoundType	An audio representation.	*not used in UBL 2.1*
VideoType	A video representation.	*not used in UBL 2.1*
CodeType	A character string (letters, figures, or symbols) that for brevity and/or language independence may be used to represent or replace a definitive value or text of an attribute, together with relevant supplementary information.	listID listAgencyID listAgencyName listName listVersionID name languageID listURI listSchemeURI
DateTimeType	A particular point in the progression of time, together with relevant supplementary information.	*format (not used in UBL 2.1)*
DateType	One calendar day according the Gregorian calendar.	
TimeType	An instance of time that occurs every day.	
IdentifierType	A character string to identify and uniquely distinguish one instance of an object in an identification scheme from all other objects in the same scheme, together with relevant supplementary information.	schemeID schemeName schemeAgencyID schemeAgencyName schemeVersionID schemeDataURI schemeURI
IndicatorType	A list of two mutually exclusive Boolean values that express the only possible states of a property.	format
MeasureType	A numeric value determined by measuring an object using a specified unit of measure.	**unitCode (required)** unitCodeListVersionID
NumericType	Numeric information that is assigned or is determined by calculation, counting, or sequencing. It does not require a unit of quantity or unit of measure.	format
ValueType	Numeric information that is assigned or is determined by calculation, counting, or sequencing. It does not require a unit of quantity or unit of measure.	*not used in UBL 2.1*
PercentType	Numeric information that is assigned or is determined by calculation, counting, or sequencing and is expressed as a percentage. It does not require a unit of quantity or unit of measure.	format

UBL Unqualified Data Type	Definition	Attributes
RateType	A numeric expression of a rate that is assigned or is determined by calculation, counting, or sequencing. It does not require a unit of quantity or unit of measure.	format
QuantityType	A counted number of non-monetary units, possibly including a fractional part.	unitCode unitCodeListID unitCodeListAgencyID unitCodeListAgencyName
TextType	A character string (i.e. a finite set of characters), generally in the form of words of a language.	languageID languageLocaleID
NameType	A character string that constitutes the distinctive designation of a person, place, thing, or concept.	languageID languageLocaleID

Some UBL BBIEs have data type qualifications based on the unqualified UBL types. These qualified types are all code types, and their definitions are the mechanism whereby a specific set of values is associated with each code.

UBL data type qualifications are expressed formally in an OASIS [**CVA**] (Context/Value Association) file contained in the `cva` directory of the 2.1 distribution.

`cva/UBL-DefaultDTQ-2.1.cva`

The specification of the CVA mechanism and format is maintained by the OASIS Code List Representation Technical Committee.

A human-readable version is provided in an accompanying HTML file, which also serves as primary documentation on the UBL codes defined as qualified data types.

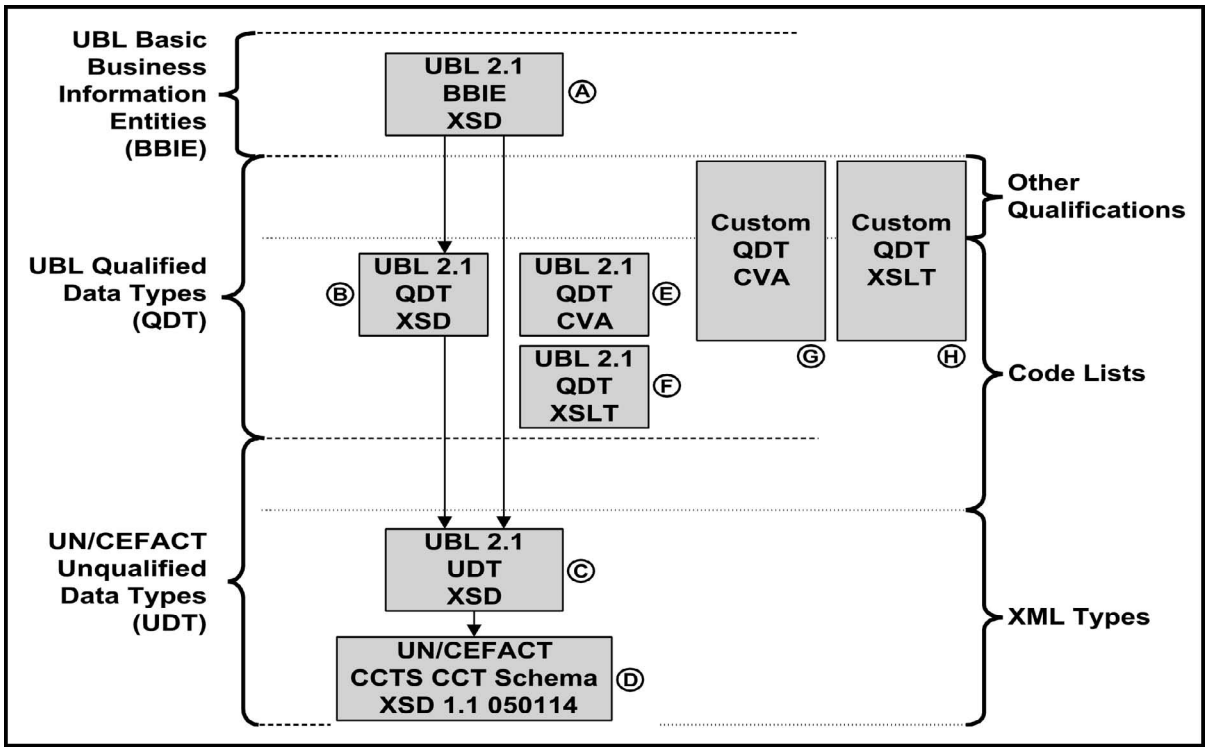
`cva/UBL-DefaultDTQ-2.1.html`

The `val` directory contains the predefined CVA associations compiled into an XSLT file, `UBL-DefaultDTQ-2.1.xsl`, which is used in the recommended two-phase validation process to perform a check of code list values. See Appendix F, *UBL 2.1 Code Lists and Two-phase Validation (Non-Normative)* for a description of this process.

`val/UBL-DefaultDTQ-2.1.xsl`

The UBL 2.1 approach to data type qualification is illustrated in the following diagram.

Figure E.1. Data Type Qualification in UBL 2.1



In UBL 2.1, the schema library of common basic components (basic information entities or BBIEs, **(A)** in the diagram) is based on a combination of the data types defined in the file of UBL 2.1 qualified data types **(B)** and the data types defined in a file of UBL 2.1 unqualified data types **(C)**. The latter inherits the data type definitions in the UN/CEFACT CCTS CCT schema module Ver. 1.1 050114 **(D)**. The UBL 2.1 CVA file **(E)** controls the creation of the UBL 2.1 XSLT stylesheet **(F)** used in validation. While this XSLT file, `UBL-2.1-DefaultDTQ.xsl`, can, in theory, apply to qualified data type qualifications in general (such as field length restrictions and value range restrictions), the version of this file included in the UBL 2.1 release contains only code list values.

The two remaining boxes on the right in the diagram illustrate that users can add further data type qualifications if desired by preparing a custom CVA **(G)** and creating a custom XSLT file **(H)** to replace the default CVA and XSLT stylesheet provided in the UBL 2.1 distribution.

Users intending to prepare a custom CVA should note that `cva/UBL-DefaultDTQ-2.1.cva` contains relative URIs that expect the UBL 2.0 code lists from the UBL 2.0 Update Package in a sibling directory named `os-UBL-2.0`. This is irrelevant to users of the precompiled `val/UBL-DefaultDTQ-2.1.xsl` file contained in the UBL 2.1 package, but users wishing to create their own CVA file must first install the UBL 2.0 release and then the UBL 2.0 update. To properly install the update, first download and install the original UBL 2.0 release:

http://docs.oasis-open.org/ubl/os-UBL-2.0.zip

Then download and install the UBL 2.0 update:

http://docs.oasis-open.org/ubl/os-UBL-2.0-update-delta.zip

Complete installation instructions can be found in the update package. As indicated above, the `os-UBL-2.0` directory thus created must be a sibling to the directory created by installing the UBL 2.1 package.

Appendix F UBL 2.1 Code Lists and Two-phase Validation (Non-Normative)

F.1 Introduction

Code lists—the sets of codes such as “FR” and “USD” that are used to specify countries, currencies, and so on—play an important role in UBL, just as they do in all electronic business messaging schemes. By default, UBL uses several lists of standard codes published by agencies such as ISO and UN/CEFACT, as well as various codes that are specific to UBL.

In UBL 1.0 (2004), standard and default code list values were enumerated directly in the UBL schemas. This allowed all UBL 1.0 instances to be validated in a single pass using generic XML XSD (W3C Schema) processors. However, the specification of the default values directly in the schemas also made it difficult to modify the code lists to suit individual trading partner relationships and impossible to extend the list of allowable code list values while still using the standard UBL schemas as published by OASIS.

To give users maximum flexibility in configuring and updating UBL code lists without changing the standard UBL schemas, UBL 2.0 introduced a two-phase validation model that has now been fully implemented in UBL 2.1. In the first phase, the UBL instance is checked for structure and vocabulary against a standard UBL schema using a generic schema validator (or custom-built software performing the same function). This is exactly the same procedure used for validation in UBL 1.0, except that the schemas do not contain hardwired code list values. Then in an added second validation (or verification) phase, code list values in the instance are checked against values obtained from external code list configuration files using an XSLT 1.0 processor driven by an XSLT 1.0 stylesheet. The default code list values assumed by the UBL 2.1 specification are expressed as data type qualifications in a file named `UBL-2.1-DefaultDTQ.xsl` located in the `val` directory, as described in more detail below. Publicly available tools were used to create the XSL file using the methodology described in the “Validation” section of [**Customization**], the *UBL Guidelines for Customization*.

Separating the checking of structure and vocabulary from the checking of code values allows trading partners to easily and precisely specify code list subsets and extensions and to apply them not just to individual UBL document types but also to particular elements and subtrees within UBL document instances. Another way to say this is that the the UBL code list methodology allows different versions of the same code list to be used in different document contexts. Thus, for example, a business in Canada might agree with a business in the United States to use a set of code list configuration files that allow the Buyer to be associated with either a U.S. state or a Canadian province but restrict the Seller to just U.S. states—that is, to apply a code list subset containing state and province codes in one place in a document instance and a different code list subset containing just state codes in another place in the instance.

F.2 Default Validation Setup

To facilitate the processing of UBL 2.1 instances using the two-phase method, an “out-of-the-box” collection of open-source software that can be used to demonstrate default validation of UBL 2.1 documents is included in the `val` directory of this release package. The validation harness assumes a Linux or Windows system with no currently installed XML or XSLT processing software.

The Java Runtime Environment (JRE) 1.5 or later is required to use the programs in the `val` directory; JRE versions below 1.5 will throw an error from the `xjparse.jar` module used to invoke the xerces schema parser. If necessary, download and install the latest JRE from the following location before continuing:

http://www.java.com/en/download/manual.jsp

To demonstrate UBL 2.1 default validation:

1. Change to the `val` directory.

2. From within that directory, enter the test command

   `test.bat` (Windows)

   or

   `sh test.sh` (Linux)

   The output, which is explained in the next section, should resemble the output shown in Figure D.1 (the spacing has been manually adjusted to make the output easier to read).

   Figure F.1. Validation test output

```
############################################################
Validating order-test-good.xml
############################################################
============== Phase 1: XSD schema validation ==============
No schema validation errors.
============ Phase 2: XSLT code list validation ============
No code list validation errors.
############################################################
Validating order-test-bad1.xml
############################################################
============== Phase 1: XSD schema validation ==============
Attempting validating, namespace-aware parse
Error:file:///c:/d/ubl/2/val/order-test-bad1.xml:48:23:cvc-complex-type.2.4.a:
Invalid content was found starting with element 'cbc:ChannelCod'.
One of '{"urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2":ChannelCode,
"urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2":Channel,
"urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2":Value}' is expected.
Parse succeeded (0.822) with 1 error and no warnings.
############################################################
Validating order-test-bad2.xml
############################################################
============== Phase 1: XSD schema validation ==============
No schema validation errors.
============ Phase 2: XSLT code list validation ============
Value supplied ' LA ' is unacceptable for codes identified by 'ChannelCodeType'
in the context: cbc:ChannelCode
Processing terminated by xsl:message at line 18
```

3. From within the `val` directory, you can now validate any UBL document against the UBL 2.1 schemas by executing commands of the form

   `validate<`*`ubl-schema`*`> <`*`ubl-document`*`>`

   where <*`ubl-document`*> is the path of a document to be validated and <*`ubl-schema`*> is the path of the UBL 2.1 schema for that document type (Order, Invoice, etc.). For example, the scripts `val/testsamples.bat` and `val/testsamples.sh` show this process being used to validate the sample XML instances in the `xml` directory.

F.3 Discussion of the Default Validation Test

The test output displayed above demonstrates the default validation process with three test files: a valid UBL Order (`val/order-test-good.xml`); a UBL Order containing a bad (misspelled) element (`val/order-test-bad1.xml`); and a UBL Order that is schema-valid but contains an illegal code list value (`val/order-test-bad2.xml`). The file `val/test.bat` (Windows) or `val/test.sh` (Linux) is used to run the script `val/validate.bat` or `val/validate.sh` against each of the test files.

The first run using `order-test-good.xml` demonstrates both phases of the default validation process running normally. In the first phase, a standard W3C Schema (XSD) validator, xerces, is invoked from

`val/w3cschema.bat` (or `val/w3cschema.sh`) to validate the specified UBL document (`.xml`) against the specified UBL 2.1 runtime schema (`.xsd`). Since the input is a valid UBL Order, the output of the first phase simply indicates that the file is valid against the given Order schema.

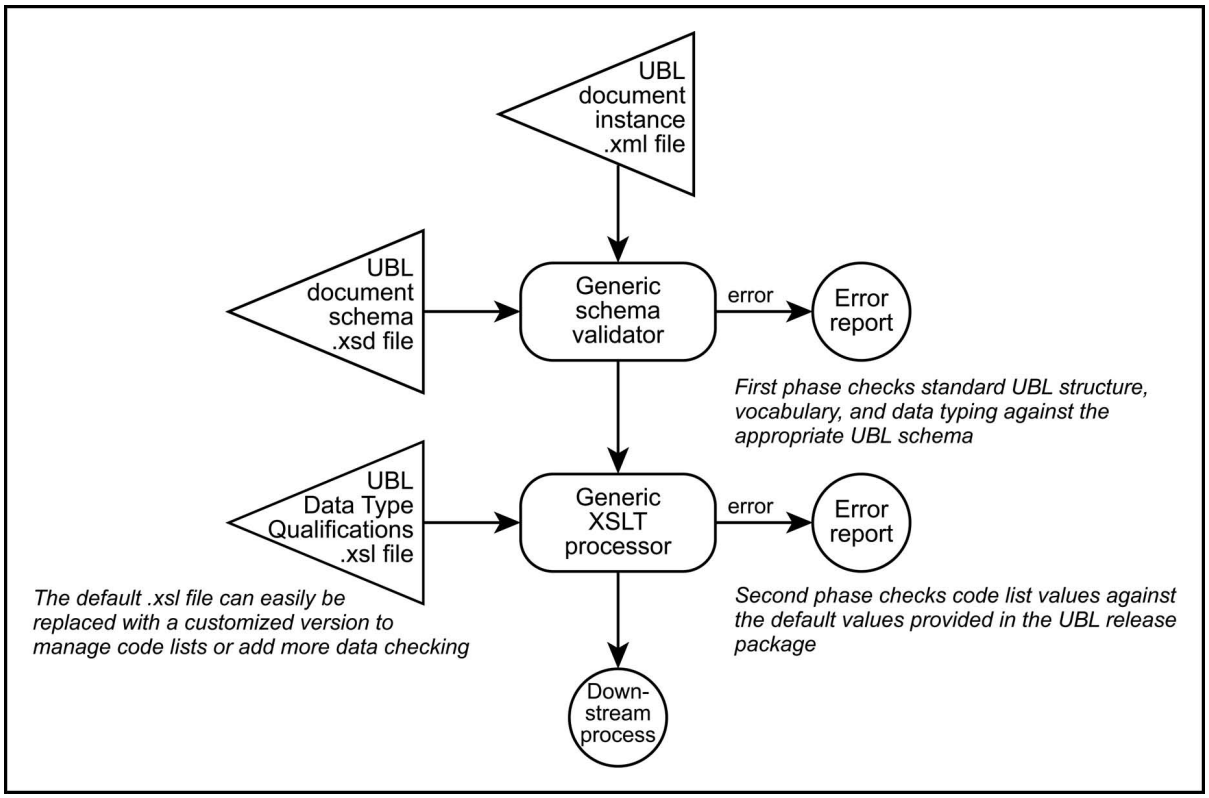
The second phase of validation uses a standard XSLT 1.0 engine, saxon, to verify that the values of various codes used in the UBL document to be tested (currency codes, packaging types, etc.) are valid in terms of the default UBL 2.1 code list values specified in `val/UBL-DefaultDTQ-2.1.xsl`. Here the output line "No code list validation errors" from the `validate` script indicates that the saxon run (invoked from `val/xslt.bat` or `val/xslt.sh`) finds no illegal code values in the document.

The second run shows what happens when the input document (`order-test-bad1.xml`) contains an actual structure or vocabulary error, in this case due to omission of the trailing "e" from the element named `cbc:ChannelCode`. When the xerces parser encounters the malformed element name, it emits the error message shown in the example, and the `validate` script reacts to a non-zero status code from `w3cschema.bat` (or `w3cschema.sh`) by terminating the validation process.

In the third run, the input document `order-test-bad2.xml` is structurally valid according to the Order schema, but it contains an illegal code list value (the ChannelCode "AL" for cell phone has been mistyped as "LA"). Thus it passes the first phase when tested against the schema but fails the second phase when tested against `val/UBL-DefaultDTQ-2.1.xsl`.

To summarize, input documents are checked in the first validation phase for correctness of structure and vocabulary, using the constraints expressed in the appropriate UBL schema, and then they are checked in the second phase for correctness of default code list values, using the default constraints expressed in the XSLT file `UBL-DefaultDTQ-2.1.xsl`. This process is illustrated in the following diagram.

Figure F.2. Two-phase Default UBL 2.1 Validation



It should be clear from the foregoing that the second phase of the default validation process can safely be omitted if it is considered unnecessary to check code list values. However, the reverse is not true; the second phase depends for correct operation on a prior check for structural validity, and therefore it will not give reliable results if run in the absence of the first (schema) validation phase.

F.4 Customizing the Default XSLT File

The validation framework provided in the `val` directory can be used to implement code list changes, define variant code lists to fit specific trading partner agreements, or associate different versions of the same code list with different parts of the same UBL document by substituting a custom process (be it XSLT or some other language or process) for the default `UBL-DefaultDTQ-2.1.xsl` provided in the UBL 2.1 distribution. This allows extensive code list management without the need to change the standard UBL 2.1 schemas. Schematron-based techniques for generating a custom XSLT file to take the place of `UBL-DefaultDTQ-2.1.xsl` are explained in [**CVA**] and [**Customization**]. See also Appendix E, *Data Type Qualifications in UBL (Non-Normative)* for more about UBL data type qualifications.

Since XSLT is a very powerful general-purpose XML transformation tool, the same framework can be extended to perform fairly sophisticated business rule checking by manually coding additional logic into the XSLT file that drives the second validation phase. Such modification is beyond the scope of the customization methodologies associated specifically with UBL, but a business analyst willing to perform XSLT programming can use this mechanism to offload a large proportion of input filtering from the backend business application to a simpler input processing area. Additional XSLT scripts can be added to extract logical subtrees of incoming UBL documents for allocation to different downstream processes and to perform even more extensive front-end processing.

F.5 Sources for the Default Validation Framework

Components of several freely available software distributions were used to create the `val` directory. Sources are given below so that these components can be updated as later releases become available.

- The files `val/resolver.jar` and `val/xercesImpl.jar` are from the xerces-j 2.8.0 binary distribution at

  http://archive.apache.org/dist/xml/xerces-j/Xerces-J-bin.2.8.0.zip

- The file `val/xjparse.jar` (renamed from `xjparse-1.0.jar`) is from the xjparse 1.0 distribution at

  http://nwalsh.com/java/xjparse/

- The file `val/saxon.jar` is from the saxon 6.5.5 distribution at

  http://prdownloads.sourceforge.net/saxon/saxon6-5-5.zip

- The file `val/UBL-DefaultDTQ-2.1.xsl` was created using the Schematron implementation of CVA files for validation at

  http://www.CraneSoftwrights.com/resources/ubl/#cva2sch

F.6 Code Lists Included in UBL 2.1

The code lists included in the UBL 2.1 distribution use an OASIS Standard XML format for code lists called [**genericode**]. Each code list in the distribution occupies its own genericode file. Documentation on the UBL code lists is contained in a generated report file:

`cva/UBL-DefaultDTQ-2.1.html`

The code list files in UBL 2.1 are divided into two subdirectories, `cl/gc/default` and `cl/gc/special-purpose`.

F.6.1 cl/gc/default

The code lists in the `cl/gc/default` directory contain the default code values represented in `UBL-DefaultDTQ-2.1.xsl`. A second-phase code list check using an unmodified version of the test setup from this distribution as described above will verify all occurrences of code values from these lists against the values specified in the `cl/gc/default` directory. These are the code lists expected to be used in most application contexts.

```
cl/gc/default/AllowanceChargeReasonCode-2.1.gc
cl/gc/default/BinaryObjectMimeCode-2.1.gc
cl/gc/default/ChannelCode-2.1.gc
cl/gc/default/CurrencyCode-2.1.gc
cl/gc/default/PackagingTypeCode-2.1.gc
cl/gc/default/PaymentMeansCode-2.1.gc
cl/gc/default/ReceiptAdviceTypeCode-2.1.gc
cl/gc/default/TransportEquipmentTypeCode-2.1.gc
cl/gc/default/TransportModeCode-2.1.gc
cl/gc/default/UnitOfMeasureCode-2.1.gc
```

F.6.2 cl/gc/special-purpose

This directory contains genericode versions of code lists that are used only in certain application contexts. They are not included in the `UBL-DefaultDTQ-2.1.xsl` file included in this distribution, but are provided here in the `cl/gc/special-purpose` directory to make them available for incorporation into custom XSLT scripts.

The files in this directory are as follows:

```
cl/gc/special-purpose/AdjustmentReasonCode-2.1.gc
cl/gc/special-purpose/CollaborationPriorityCode-2.1.gc
cl/gc/special-purpose/ComparisonDataSourceCode-2.1.gc
cl/gc/special-purpose/DataSourceCode-2.1.gc
cl/gc/special-purpose/DisplayTacticTypeCode-2.1.gc
cl/gc/special-purpose/ExceptionStatusCode-2.1.gc
cl/gc/special-purpose/ForecastPurposeCode-2.1.gc
cl/gc/special-purpose/ForecastTypeCode-2.1.gc
cl/gc/special-purpose/LanguageCode-2.1.gc
cl/gc/special-purpose/MiscellaneousEventTypeCode-2.1.gc
cl/gc/special-purpose/PerformanceMetricCriterionCode-2.1.gc
cl/gc/special-purpose/PromotionalEventTypeCode-2.1.gc
cl/gc/special-purpose/ResolutionCode-2.1.gc
cl/gc/special-purpose/RetailEventStatusCode-2.1.gc
cl/gc/special-purpose/RevisionStatusCode-2.1.gc
cl/gc/special-purpose/StatusCode-2.1.gc
cl/gc/special-purpose/SupplyChainActivityTypeCode-2.1.gc
cl/gc/special-purpose/ThresholdValueComparisonCode-2.1.gc
cl/gc/special-purpose/TimeFrequencyCode-2.1.gc
cl/gc/special-purpose/TransportationStatusCode-2.1.gc
```

F.7 Code List Agency Identifiers

The code list information provided in the form of genericode files in this distribution (see Section F.6, “Code Lists Included in UBL 2.1”) comes from two sources: the OASIS UBL Technical Committee and UN/CEFACT, the Centre for Trade Facilitation and Electronic Business of the UN Economic Commission for Europe (UNECE). The custodial agency responsible for each source is identified in the corresponding genericode files as shown in the following examples.

```
<Agency>
  <LongName xml:lang="en">OASIS Universal Business Language</LongName>
  <Identifier>UBL</Identifier>
</Agency>
```

```
<Agency>
  <LongName xml:lang="en">United Nations Economic Commission for Europe</LongName>
  <Identifier>6</Identifier>
</Agency>
```

The “6” identifier for UNECE is from the code list "Responsible Agency Code" most recently published at

http://www.unece.org/trade/untdid/d11a/tred/tred3055.htm

Appendix G UBL 2.1 Example Document Instances (Non-Normative)

The `xml` directory of this distribution contains a number of sample UBL documents that can be used for testing purposes. The `testsamples.bat` batch file and the `testsamples.sh` script in the `val` directory of this distribution can be used to demonstrate the validity of these examples in Windows and Linux operating environments. See Appendix F, *UBL 2.1 Code Lists and Two-phase Validation (Non-Normative)* for a general discussion of UBL validation methodology. For convenience, those examples that relate specifically to a particular document type are linked from the description of that type in Section 3.1, "UBL 2.1 Document Schemas".

Example instances containing extensions

xml/MyTransportationStatus.xml
xml/UBL-Invoice-2.0-Enveloped.xml

Example instances related to signatures (see Section 5.4, "Digital Signature Examples")

xml/UBL-Invoice-2.0-Detached-Signature.xml
xml/UBL-Invoice-2.0-Detached.xml
xml/UBL-Invoice-2.0-Enveloped.xml

Example instances with unconventional use of namespace bindings

xml/UBL-Invoice-2.0-Example-NS1.xml
xml/UBL-Invoice-2.0-Example-NS2.xml
xml/UBL-Invoice-2.0-Example-NS3.xml
xml/UBL-Invoice-2.0-Example-NS4.xml

Example instances of different versions of certain document types

xml/UBL-CreditNote-2.0-Example.xml
xml/UBL-CreditNote-2.1-Example.xml
xml/UBL-DebitNote-2.1-Example.xml
xml/UBL-DespatchAdvice-2.0-Example.xml
xml/UBL-ExceptionCriteria-2.1-Example.xml
xml/UBL-ExceptionNotification-2.1-Example.xml
xml/UBL-Forecast-2.1-Example.xml
xml/UBL-ForecastRevision-2.1-Example.xml
xml/UBL-ForwardingInstructions-2.0-Example-International.xml
xml/UBL-FreightInvoice-2.1-Example.xml
xml/UBL-FulfilmentCancellation-2.1-Example.xml
xml/UBL-GoodsItemItinerary-2.1-Example.xml
xml/UBL-InstructionForReturns-2.1-Example.xml
xml/UBL-InventoryReport-2.1-Example.xml
xml/UBL-Invoice-2.0-Example.xml
xml/UBL-Invoice-2.1-Example.xml
xml/UBL-Invoice-2.1-Example-Trivial.xml
xml/UBL-Order-2.0-Example-International.xml
xml/UBL-Order-2.0-Example.xml
xml/UBL-Order-2.1-Example.xml
xml/UBL-OrderCancellation-2.1-Example.xml
xml/UBL-OrderChange-2.1-Example.xml
xml/UBL-OrderResponse-2.1-Example.xml

xml/UBL-OrderResponseSimple-2.0-Example.xml
xml/UBL-OrderResponseSimple-2.1-Example.xml
xml/UBL-PerformanceHistory-2.1-Example.xml
xml/UBL-ProductActivity-2.1-Example-1.xml
xml/UBL-ProductActivity-2.1-Example-2.xml
xml/UBL-ProductActivity-2.1-Example-3.xml
xml/UBL-Quotation-2.0-Example.xml
xml/UBL-Quotation-2.1-Example.xml
xml/UBL-ReceiptAdvice-2.0-Example.xml
xml/UBL-Reminder-2.1-Example.xml
xml/UBL-RemittanceAdvice-2.0-Example.xml
xml/UBL-RequestForQuotation-2.0-Example.xml
xml/UBL-RequestForQuotation-2.1-Example.xml
xml/UBL-RetailEvent-2.1-Example.xml
xml/UBL-SelfBilledCreditNote-2.1-Example.xml
xml/UBL-Statement-2.0-Example.xml
xml/UBL-StockAvailabilityReport-2.1-Example.xml
xml/UBL-TradeItemLocationProfile-2.1-Example.xml
xml/UBL-TransportationStatus-2.1-Example.xml
xml/UBL-TransportationStatusRequest-2.1-Example.xml
xml/UBL-TransportExecutionPlan-2.1-Example.xml
xml/UBL-TransportExecutionPlanRequest-2.1-Example.xml
xml/UBL-TransportProgressStatus-2.1-Example.xml
xml/UBL-TransportProgressStatusRequest-2.1-Example.xml
xml/UBL-TransportServiceDescription-2.1-Example.xml
xml/UBL-TransportServiceDescriptionRequest-2.1-Example.xml
xml/UBL-Waybill-2.0-Example-International.xml

Appendix H Alternative Representations of the UBL 2.1 Schemas (Non-Normative)

UBL 2.1 continues the practice, adopted at the beginning of the UBL effort, of creating its normative XML specifications using W3C Schema (XSD) syntax. Included in this release are two additional alternative specifications of the same content: a set of UBL 2.1 ASN.1 modules and a set of UBL 2.1 RELAX NG (compact syntax) schemas. These alternative representations are technically non-normative, but both are generated directly from the XSD and, with the exception of the UBL 2.1 digital signature extension (see Section 5.3, "UBL Extension for Enveloped XML Digital Signatures"), both are intended to implement the same document instance constraints.

H.1 ASN.1 UBL 2.1 Specification

The UBL ASN.1 specification linked below provides an alternative schema definition for UBL documents in accordance with ITU-T X.680-X.693 [**ASN.1**]. The UBL ASN.1 specification defines the same UBL documents as the UBL XSD schemas that constitute the normative definitions of valid UBL documents. The UBL ASN.1 XML specification enables ASN.1 tools to be used for UBL transfers, and in conjunction with the ASN.1 Packed Encoding Rules, it provides a specification for an efficient binary encoding of UBL messages.

The zip archive below contains the ASN.1 modules corresponding to the UBL 2.1 document schemas as individual text files. The ASN.1 modules were created using a tool from OSS Nokalva [http://www.oss.com/] that conforms to ITU-T Recommendation X.694 | ISO/IEC 8825-5 for converting XSD Schema to ASN.1.

UBL 2.1 ASN.1 Modules
`asn/ASN.1-UBL-2.1-text.zip`

After conversion from XSD, the generated ASN.1 was formatted by the PrettyPrint tool at the ASN.1 Information Site [http://asn1.elibel.tm.fr/] to produce the following HTML documentation file.

UBL 2.1 ASN.1 Specification
`asn/ASN.1-UBL-2.1.html`

H.2 UBL 2.1 RELAX NG Schemas

[**RELAX NG**] (compact syntax) versions of the UBL schemas contributed by Crane Softwrights [http://cranesoftwrights.com/] and used here by permission are located in the `rnc` directory. The Crane package includes RELAX NG schemas for both UBL 2.0 and 2.1, as detailed in the related documentation.

`rnc/readme-rnc.html`

The UBL 2.1 RELAX NG schema for each UBL 2.1 document type is linked from the description of that document type in Section 3.1, "UBL 2.1 Document Schemas".

Appendix I The Open-edi reference model perspective of UBL (Non-Normative)

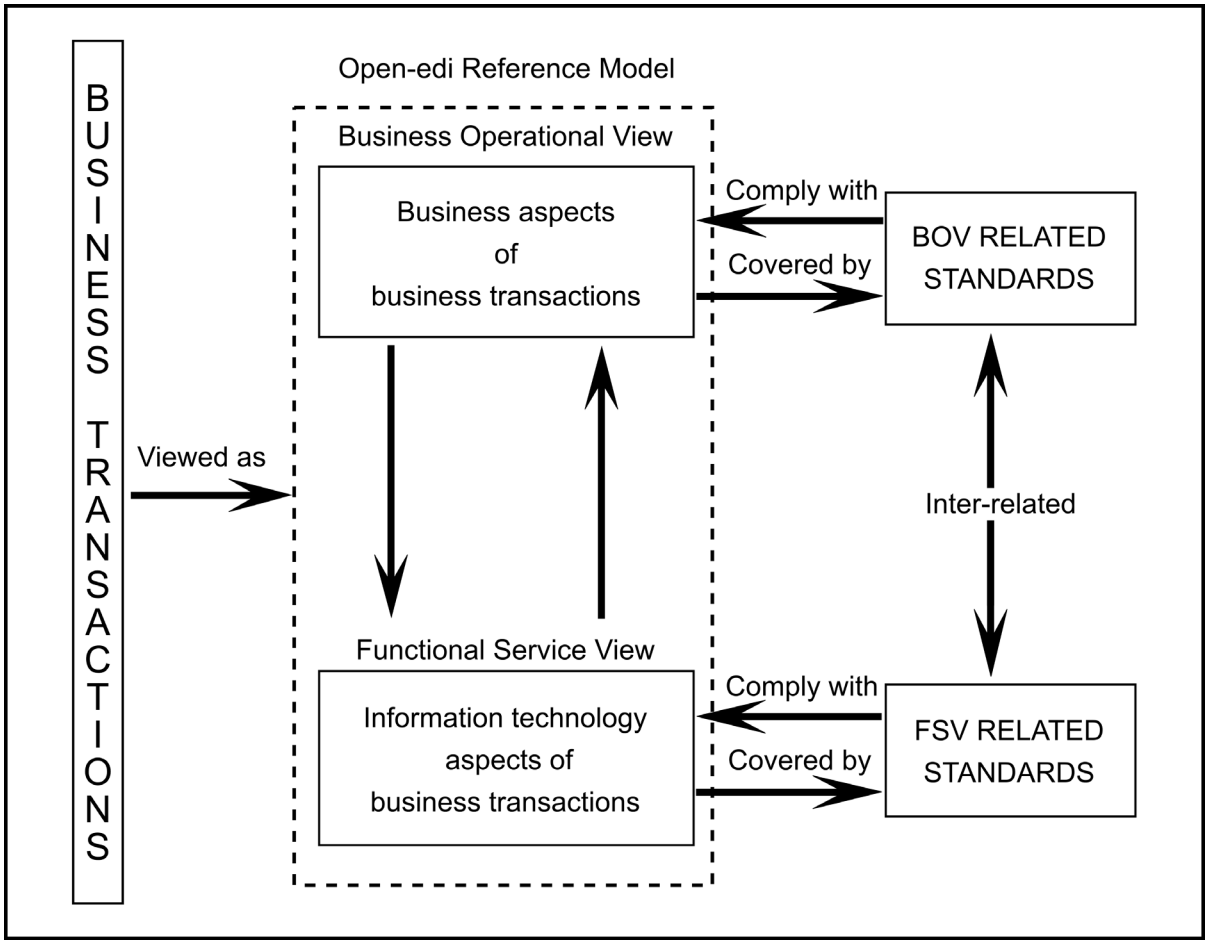
ISO/IEC 14662:2010 Information technology - Open-edi reference model [**Open-edi**] has been developed primarily in order to provide standards required for the inter-working of organizations, through interconnected information technology systems. Open-edi lowers barriers to electronic data interchange by introducing standard business scenarios and the necessary services to support them.

The Open-edi Reference Model identifies the required standards for Open-edi and provides a reference for those standards by defining the basic concepts used to develop them.

Figure I.1, “Open-edi Overview” depicts two views to describe the relevant aspects of business transactions:

- the Business Operational View (BOV);
- the Functional Service View (FSV).

Figure I.1. Open-edi Overview



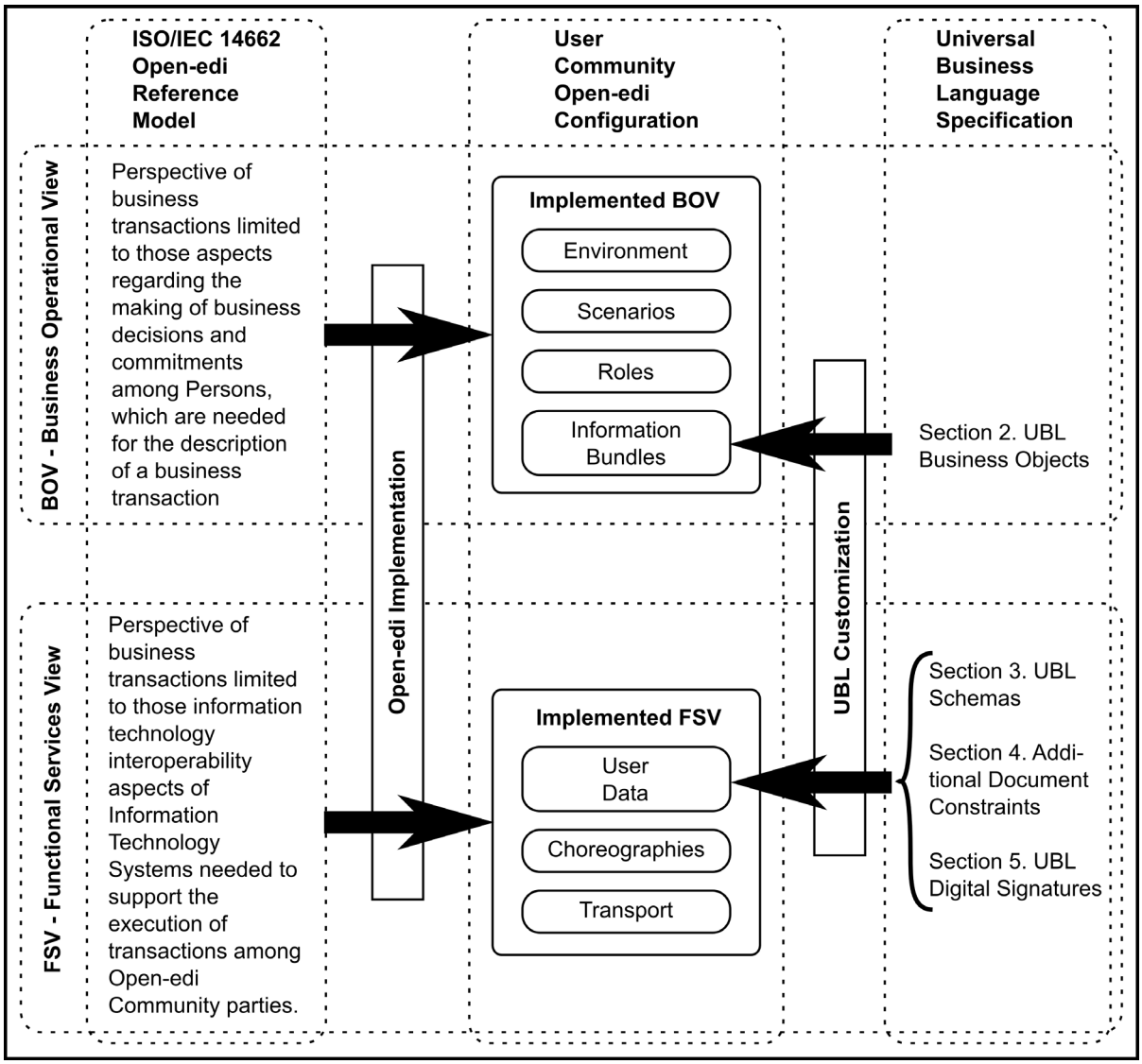
The BOV addresses the aspects of the semantics of business data in business transactions and associated data interchanges which apply to the business needs of Open-edi. The BOV-related standards are tools and rules by which users, who understand the operating aspects of a business domain, may create scenarios.

The FSV addresses the supporting services meeting the mechanistic needs of Open-edi, focusing on information technology aspects of functional capabilities, service interfaces and protocols. Using the concepts of Open-edi, UBL 2.1 provides a generic Open-edi Configuration that an Open-edi Community may customize with their own requirements to implement their own Open-edi Configuration.

ISO/IEC 15944-20 Information technology - Business operational view - Linking business operational view to functional service view [**BOV-FSV**] presents the relationships linking the BOV with the FSV.

Figure I.2, "Open-edi Application" illustrates how the two normative deliverables of UBL, the semantic components and the XML schemas, align respectively with the BOV and FSV views of the Open-edi Reference Model.

Figure I.2. Open-edi Application



Section 2, "UBL 2.1 Business objects" provides the configuration's BOV with a suite of normative business objects and associated semantics from which the community selects the semantic components needed in an information bundle. An information bundle describes the semantics of the recorded information to be exchanged between Open-edi Support Infrastructures servicing Decision Making Applications. The community's configuration combines these information bundles with their identified scenarios and roles governed by internal and external influences of the environment.

Section 3, "UBL 2.1 Schemas" and Section 4, "Additional Document Constraints" provides the configuration's FSV with a set of corresponding normative XML schemas and document instance rules constraining

the expression of the business objects in user data. One translates the semantic component values into a transfer syntax from the information bundle specification as a set of recorded information. It is the UBL XML syntax for the sets of recorded information defined by the information bundles that are exchanged between Parties.

Section 5, “UBL Digital Signatures” provides the configuration's FSV with a normative schema fragment suitable for including profiles of advanced digital signatures in user data.

The other aspects of the implemented BOV and implemented FSV of the community's Open-edi Configuration are governed by influences outside of the scope of UBL. Those aspects guide the community in customizing UBL to suit their requirements, as outlined in Section A.5, “UBL Customization”.

Appendix J Acknowledgements (Non-Normative)

The OASIS UBL Technical Committee thanks Altova for its contribution of XML Spy licenses for use in UBL schema design; Sparx Systems for its contribution of Enterprise Architect licenses for use in developing UML content models; Syncro Soft for its contribution of oXygen licenses used in DocBook authoring of UBL documentation; RenderX for its contribution of XEP licenses used in generating PDF documents from DocBook originals; SRDC for developing iSurf eDoCreator (supported by European Commission FP7 ICT-213031 iSurf Project) and providing technical assistance in its use as an online repository and editing environment for UBL 2.1 document models and the generation of UBL schemas; OSS Nokalva for its contribution of ASN.1 UBL 2.1 specifications; and Crane Softwrights for permission to include a copy of their publicly-available HTML reports and UBL 2.1 RELAX NG schema resources.

The following persons and companies participated as members of the OASIS UBL Technical Committee during the four years of its development (2008–2012).

Inigo Barreira, iZenpe
Roger Bass, Traxian
Oriol Bausa Peris, Invinet Sistemes 2003
Kenneth Bengtsson, Alfa1lab
Georg Birgisson, Document Engineering Services
Peter Borresen, Denmark Ministry of Science, Technology and Innovation
Jon Bosak, Sun Microsystems
Mikkel Brun, Tradeshift Network
Arianna Brutti, ENEA UTT PMI
Andrea Caccia, AITI (Associazione Italiana Tesorieri de Impresa)
Manuel Cano, Nexus IT
Sally Chan, Boeing
William Chan, Individual
Roberto Cisternino, Individual
Anthony Coates, Document Engineering Services
Gary Cornelius, CSW Group
Mavis Cournane, Cognitran
Robin Cover, OASIS
Eduardo Criado Albuixech, Eurobits Technologies
Juan Cruellas, Departamento de Arquitectura de Computadores, Univ Politecnica de Cataluna
Piero De Sabbata, ENEA UTT PMI
Michael Dill, Individual
Asuman Dogac, Middle East Technical University
Kees Duvekot, RFS Holland Holding
Pim van der Eijk, Sonnenglanz Consulting
David Fitzpatrick, Booz Allen Hamilton
Martin Forsberg, Swedish Association of Local Authorities and Regions
Robert Glushko, University of California, Berkeley
Arturo Gonzalez Mac Dowell, Eurobits Technologies
Stephen Green, SystML
Michael Grimley, US Department of Defense (DoD)
Eduardo Gutentag, Sun Microsystems
Betty Harvey, ECC
Anne Hendry, Individual
Hideki Hiura, Justsystems
Ken Holman, Crane Softwrights Ltd.
Naomasa Hosoda, NEC
Ismar Huskic, Document Engineering Services
Julian Inza, Eurobits Technologies

Akihiro Kawauchi, Individual
Kyung-In Kim, Korea Institute for Electronic Commerce (KIEC)
Sung Hyuk Kim, Individual
Stig Korsgaard, Danish Bankers Association
Ram Kumar, Individual
John Larmouth, Individual
Thomas Lee, University of Hong Kong
Thomas Love, efoil
Luis Martin-Santos, Gaia Net Exchange
Tim McGrath, Document Engineering Services
Brais Mendez Ferreiro, Sociedad de Explotacion de Redes Electronicas y Servicios (SERES)
Garret Minakawa, Oracle
Tuncay Namli, Individual
Yasuyuki Nishioka, PSLX consortium
Dave Nurse, CSW Group
Cagdas Ocalan, Middle East Technical University
Mark Palmer, NIST
Klaus Pedersen, OIOXML eBusiness Standardization Group
Sue Probert, Individual
Sven Rasmussen, Denmark Ministry of Science, Technology and Innovation
Zarella Rendon, PTC
Yukinori Saito, ERP Research Corporation
Sacha Schlegel, Individual
Andrew Schoka, Individual
Mark Seaborne, PicoForms
Jose Silva, Individual
Ali Sinaci, Middle East Technical University
Kumar Sivaraman, Oracle
Enric Staromiejski, SOM Team
Paul Thorpe, OSS Nokalva
Karsten Tolle, Document Engineering Services
Juerg Tschumperlin, New Zealand Ministry of Education
Fulya Tuncer, Middle East Technical University
Kenneth Vaughn, Individual
Vito Vavalli, AITI-Associazione Italiana Tesorieri de Impresa
Audun Vennesland, SINTEF
Catherine Williams, PISCES
Nigel Wooden, ACORD
Marcelo Yarzabal, Eurobits Technologies
Patrick Yee, University of Hong Kong
Arif Yildirim, Revenue Administration of Turkey
Peter Yim, CIM Engineering
Pine Zhang, UOML Alliance