FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

A User Guide for UN/CEFACT's Process and Data Modeling Methodologies

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur/in

im Rahmen des Studiums

WirtschaftsInformatik

eingereicht von

Sonia Sheethal Madtha
Matrikelnummer 0248814

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuer: Univ.-Ass. Dipl.-Ing. Mag. Dr. Marco Zapletal

Wien, October 1, 2013 _____ _____

(Unterschrift Verfasser/in) (Unterschrift Betreuer)

Erklärung zur Verfassung der Arbeit

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Sonia Madtha
Wien, October 1, 2013

Abstract

B2B e-commerce has drawn increasing attention in the area of information exchange. Modeling B2B activities focuses primarily on two perspectives: one based on the format of data exchanged between trading partners and the other based on how the data is exchanged during business collaborations. The United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) has introduced UN/CEFACT's Modeling Methodology (UMM) to close the semantic and context gap, on both the process and data level in B2B. UMM is a promising standardized solution towards modeling business processes in an efficient way, by identifying business requirements and designing business collaborations between organizations from a neutral perspective. During business transactions, business documents are exchanged between trading partners. UN/CEFACT recommends using UML Profile for Core Components (UPCC) for designing business documents for the collaboration process.

Though UMM and UPCC encourage modelers to reuse artifacts, ensure the capability of defining models on a conceptual level and overcome inter-operability issues, these standards are complex and require high level of expertise for modelers. This guide is used to assist the modeler, starting from the point where business requirements are captured until the final phase where an efficient business process collaboration model and business documents are created.

This thesis covers all the aspects of UMM 2.0 and UPCC 3.0 specifications adopted by UN/CEFACT. It introduces a step-by-step modeling compendium that starts on a conceptual level and guides the modeler through the workflow of creating artifacts towards a UMM-compliant business collaboration model. Thereby, modeling techniques are suggested that help to deal with complex semantics and fasten the modeling process by reusing artifacts. Furthermore, this thesis introduces worksheets that support capturing domain knowledge from business experts. The artifacts created during the modeling process, are summarized after each milestone is reached. Finally, the modeling guide is accompanied by an example scenario that show by example how to model UMM and UPCC in an efficient way and to ensure the overall quality of the resulting business process model.

Keywords: E-Commerce, UN/CEFACT, UMM 2.0, UPCC 3.0.

Kurzfassung

Die IT-unterstützte Automatisierung von Business-to-Business (B2B) Geschäftstransaktionen erfordert die Erfassung und Analyse aller Aktivitäten, die für unternehmensübergreifende Prozesse relevant sind. Dazu kommen graphische Modellierungsansätze zum Einsatz, die vor allem auf zwei Perspektiven fokussieren: Die Datenperspektive konzentriert sich auf die semantischen Inhalte der ausgetauschten Geschäftsdokumente, während die Prozessperspektive die Abfolge des Austausches der Geschäftsdokumente beschreibt. Das United Nations Center for Trade Facilitation and Electronic Business (UN/CEFACT) hat mit UN/CEFACT's Modeling Methodology (UMM) einen Modellierungsansatz eingeführt um die semantische und kontextuelle Lücke sowohl auf Prozessebene als auch auf Datenebene zu schließen. UMM ist eine viel verspechende standardisierte Lösung zur Modellierung von Geschäftsprozessen, die es aus einer neutralen Perspektive ermöglicht, Anforderungen zu identifizieren und die Zusammenarbeit zwischen den Organisationen zu gestalten. Im Kollaborationsprozess werden Geschäftsdokumente zwischen Geschäftspartnern ausgetauscht. UN/CEFACT empfiehlt den Einsatz von UML-Profile for Core Components (UPCC) um Geschäftsdokumente, die für den Informationsaustausch in UMM benötigt werden, auf einer konzeptionellen Ebene zu entwerfen.

Obwohl UMM und UPCC Interoperabilitätsprobleme gut bewältigen, die Möglichkeit zur konzeptionellen Defintion von Dokumenten bieten und die Wiederverwendbarkeit von Artefakten ermöglichen, erfordern diese Werkzeuge ein hohes Maß an Fachwissen. Die Diplomarbeit unterstützt den Modellierer Schritt für Schritt, beginnend bei der Erfassung der Geschäftsanforderungen bis hin zur Endphase, in der effiziente Geschäftsprozess-kollaborationsmodelle und Geschäftsdokumente erstellt werden.

Diese Arbeit deckt die neuesten Aspekte der UMM 2.0 und UPCC 3.0 Spezifikationen ab, welche von UN/CEFACT standardisiert wurden. Zuerst wird die konzeptionelle Ebene grundlegend dargestellt und mit leicht verständlichen Details erkärt. Darauf aufbauend werden komplexe Zusammenhänge mit Hilfe von UML Aktivitätsdiagrammen erläutert. Dabei werden die einzelnen Prozessschritte aufgezeigt, welche für die Entwicklung eines leistungsfähigen Geschäftsprozesses notwendig sind. Effiziente Modellierungstechniken werden verwendet, um die Wiederverwendung von Artefakten gewährleisten zu können. Durch vordefinierte UMM Worksheet Vorlagen ist es möglich das Wissen der Business-Experten zu sammeln und gebündelt über Modelle hinweg wiederzuverwenden. Die Artefakte die während der Modellierung erstellt wurden, werden nach jedem erreichten Meilenstein zusammengefasst. Der gesamte Ablauf wird dabei anhand eines Beispielszenarios dargestellt.

Keywords: E-Commerce, UN/CEFACT, UMM 2.0, UPCC 3.0.

Contents

Abstract ii

Kurzfassung iii

Contents iv

List of Figures vi

List of Tables viii

1 Introduction 1
- 1.1 Electronic Data Interchange . . . 2
  - 1.1.1 Background of EDI . . . 2
  - 1.1.2 Changing Face of EDI . . . 3
  - 1.1.3 B2B Open EDI Reference Model . . . 6
- 1.2 Contribution . . . 7
- 1.3 Web Page and Teaching Supplements . . . 8

2 Related work 9
- 2.1 Business process modeling standards . . . 9
- 2.2 Business document modeling standards . . . 12
- 2.3 Other Approaches . . . 16

3 UN/CEFACT Standards for modeling e-business transactions 17
- 3.1 About UN/CEFACT . . . 17
- 3.2 Overview about UMM . . . 18
  - 3.2.1 UMM Modules . . . 20
- 3.3 Overview about UPCC . . . 21
- 3.4 Relation between UMM and UPCC . . . 24
- 3.5 UML Profile . . . 24
  - 3.5.1 Stereotypes . . . 25
  - 3.5.2 Tagged Values . . . 26

3.5.3 Constraints 27

4 Exploring the Order-From-Quote example 29

5 UMM User Guide 31
5.1 UMM Structure 31
5.1.1 Business Requirements View 32
5.1.2 Business Choreography View 32
5.1.3 Business Information View 33
5.2 Worksheets in UMM 34
5.3 Business Collaboration Model 35
5.4 Business Requirements View 37
5.4.1 Business Domain View 40
5.4.2 Business Partner View 57
5.4.3 Business Entity View 59
5.5 Business Choreography View 65
5.5.1 Business Transaction View 68
5.5.2 Business Collaboration View 84
5.5.3 Business Realization View 98
5.6 Business Information View 103

6 UPCC User Guide 105
6.1 Overview of UN/CEFACT's Core Components 105
6.2 UPCC Library 107
6.2.1 Core Component Library 110
6.2.2 Core Data Type Library 113
6.2.3 Primitive Type Library 115
6.2.4 Enumeration Type Library 117
6.2.5 Business Information Entity Library 119
6.2.6 Business Data Type Library 122
6.2.7 Business Document Library 125

7 Conclusion 129

A Worksheets 131

B Stereotype Abbreviations - UMM 143

C Stereotype Abbreviations - UPCC 147

Bibliography 151

List of Figures

1.1 Open-EDI Reference Model 7

2.1 EDIFACT Message Format 15

3.1 Evolution of UMM 19
3.2 UMM Meta Model 20
3.3 Overview of UPCC 23
3.4 Meta-Object Facility Architecture 25
3.5 Stereotype Representation 25
3.6 Representation of Tagged Values in a Class diagram 26
3.7 Tagged Values: key-value pair 27
3.8 Business Transaction Example 28

4.1 Workflow of the Order from Quote 30

5.1 Workflow - UMM and UPCC 32
5.2 Workflow - WorksheetEditor 34
5.3 Workflow - Business Collaboration Model 36
5.4 Business Collaboration Model - OrderFromQuote 37
5.5 Workflow - Business Requirement View 38
5.6 Business Requirements View - OrderFromQuote 39
5.7 Workflow - Business Domain View 42
5.8 Business Domain View - Order From Quote 43
5.9 Business Area - Procurement Sales 45
5.10 Process Area - Negotiation 48
5.11 Process Area Use Case Diagram - Negotiation 48
5.12 Business Process Use Case Diagram- Inter-organizational Process (to-be) . 49
5.13 Business Process Activity Diagram - Place Purchase Order 52
5.14 Business Process Activity Diagram - Inter-organizational Process (to-be) . . 53
5.15 Business Partner View - Order From Quote 57
5.16 Workflow - Business Partner View 57
5.17 Workflow - Business Entity View 61

5.18 Business Entity View - Order From Quote 62
5.19 Business Entity State Machine Diagram - Quote 64
5.20 Workflow - Business Choreography View 66
5.21 Business Choreography View - Order From Quote 67
5.22 Workflow - Business Transaction View 68
5.23 Business Transaction View - Place Order 69
5.24 Business Transaction Use Case - Place Order 72
5.25 Business Transaction - Place Order 76
5.26 Workflow - Business Collaboration View 84
5.27 Order From Quote- Business Collaboration Use Case 86
5.28 Business Collaboration Use Case - Tree View 86
5.29 Order From Quote - Business Collaboration Protocol 93
5.30 Business Collaboration Protocol - Tree View 94
5.31 Workflow - Business Realization View 99
5.32 Order from Quote - Business Realization Use Case 101
5.33 Business Realization View - Tree View 101
5.34 Business Information View - Tree View 103

6.1 Relationship between Core Components and Business Information Entities 106
6.2 Core Component Library - Tree View 111
6.3 Core Data Library - Tree View 114
6.4 Primitive Library - Tree View 116
6.5 Enumeration Library - Tree View 118
6.6 Business Information Entity Library - Tree View 120
6.7 Business Data Type Library - Tree View 123
6.8 Business Document Library - Quote Envelope 127

List of Tables

5.1 Business Domain View Worksheet - OrderFromQuote 41
5.2 Business Area Worksheet - Procurement/Sales 44
5.3 Process Area Worksheet - Negotiation 46
5.4 Business Process Worksheet - Place Purchase Order 51
5.5 Business Entity Worksheet - Quote 61
5.6 Business Transaction Use Case Worksheet - Place Order 73
5.7 Business Transaction - Place Order 78
5.8 Default Tagged Values - Requesting Role 82
5.9 Default Tagged Values - Responding Role 82
5.10 Business Collaboration Use Case - Order From Quote 87
5.11 Business Collaboration Protocol - Order From Quote 95

6.1 Properties of Core Components 106

A.1 Business Domain View Worksheet 131
A.2 Business Area Worksheet 132
A.3 Process Area Worksheet 133
A.4 Business Process Worksheet 134
A.5 Business Entity Worksheet 135
A.6 Business Transaction Use Case Worksheet 137
A.7 Business Transaction Worksheet 139
A.8 Business Collaboration Use Case Worksheet 140
A.9 Business Collaboration Protocol Worksheet 141

CHAPTER 1

Introduction

Once upon a time, consumers had to go to well-known market areas to acquire the things that made their life better. As time went by, retailers and marketers came up with plenty of ways of getting their customers what they wanted. Personal contact, confined groups and primary locations were some of the prominent methods through which companies used to gain trust from their customers. Due to such considerable number of business activities, companies were brought to a competitive position thereby adding value to their customers. The emphasis was mainly on the development of isolated business strategies trying to meet the market requirements. Newspapers, radio, television, magazines enabled the consumers to look for what they wanted to purchase and where they wanted to make purchases.

Along the way, wide access to the internet has brought a whirlwind change in the economy where digital networking and communication infrastructure provides a global platform, through which enterprises and individuals interact, collaborate, communicate and search for information. Since the last decade, business sectors have improved so much that the latest technology has made business trade easier and comfortable. Trading between companies is now possible through internet or internet-related technologies, referred to as electronic business (e-business). In simple words, it is about doing business electronically.

Gone are the days when business could be planned much more in advance. Enterprises tried to build a long term relationship between suppliers, providers and sales forces by contacting cheap suppliers and sales agents. However, business attitude has changed in the past few years. Business relationships are short lived and business partners and transactions are replaced every couple of years. This dynamic e-business involves spontaneous agreement made online, with no face-to-face relationships. Business strategies are expanding so rapidly that it is hard to predict the business structure after a couple of years. The companies or businesses of present and future are not based

on individual decisions, rather their success is based on the collaborative joint work of two or more business organisations. E-business, with this new evolution of trading is focused on clients and suppliers who work together, improving efficiency and performance while performing business activities and business planning [42].

The trading perspective of e-business is referred to as electronic commerce (e-commerce). While e-business is more general than e-commerce, the former conducts business activities of a business organisation over the internet while the latter covers a wide range of business activities for products and services over the internet [1]. There are different types of e-commerce. Four basic categories based on customer focus are the following: Business-to-Business (B2B) e.g. manufacturers selling to distributors; Business-to-Consumer (B2C) e.g. web shops, where goods and services from a seller are sold directly to consumers over an online store/shop; Consumer-to-Business (C2B) e.g advertisements posted by people on major websites such as Goggle; Consumer-to-Consumer (C2C) e.g. auction platforms, where one customer sells an item to another.

This user guide focuses on B2B e-commerce. The trading conducted between enterprises in B2B involves buyers and sellers. The internet acts as a channel for marketing, sales and public relation activities. Some examples are electronic invoices, purchase orders, requests for quotations, bills of lading and receiving reports. B2B offers several advantages such as better accuracy, faster order processing, lower procurement and operational costs and better coordination among sales, operations and purchasing.

1.1 Electronic Data Interchange

In general, e-commerce lets businesses stay competitive by using innovative technologies, enabling organizations and individuals to conduct business via the internet. This way, many opportunities exist for companies to expand and provide many benefits for both companies and customers. Instead of having paper documents exchanged, enterprises are provided with the opportunity to exchange business information and messages electronically. The benefit from this is less delay and better accuracy of data handling. Before e-commerce became synonymous with the internet based technology, it was also known as Electronic Data Interchange (EDI). It was the first form of e-business to initiate inter-organizational business activities. The basic idea behind EDI was to transmit messages between information systems without human intervention, thereby enabling companies to communicate easily. EDI technology has been in use for more than 40 years enabling enterprises to exchange data electronically.

1.1.1 Background of EDI

Edward A. Guilbert first presented the idea of EDI in 1948 while working at Berlin Aircraft. However, his concepts were applied in the late 1960's by different sectors

such as railroad, airlines, ocean shipping etc., and a group was subsequently organized. This group was concerned in improving the quality and studying the problems of inter-company exchanges of transportation data named Transportation Data Coordinating Company (TDDC)[2]. Through TDDC, the first EDI standards were established. Before the internet had come into the industry, EDI messages were exchanged through Value Added Networks (VAN). Although EDI proved to be efficient, it was expensive and costly to implement due to the inter-company trading, leading to a large number of EDI standards. In 1979, the American National Standards Institute (ANSI) X12 Committee [3] was created that helped move towards EDI standardization. In 1985, on the basis of ANSI X12, the United Nations chartered the United Nations Electronic Data Interchange for Administration, Commerce and Transport (UN/EDIFACT) [4][5], which developed an international standard structure for trading between business partners. The messages exchanged between business partners are defined in a specific standard format which is EDI compatible.

In the late 1990's, due to the dominant use of ANSI X12 and EDIFACT in the automotive industry, only large companies could afford setting up EDI relationships. However small and medium enterprises (SMEs) could not. One of the major reasons was that the EDI standards standardize the structure of business documents. However, business processes are not taken into consideration. Release of multiple standards and multiple versions of the same standard affected the semantics of the standard which unfortunately were not updated by all firms since the cost of updating was relatively high and most of the enterprises preferred to stick to their well-established version of standard. In order to integrate EDI concepts in any business application, one needed to have a detailed knowledge of how the standards are implemented. When a collaborative business agreement was initiated that involved large enterprises as well as SMEs, the SMEs relied on the commercial "off-the-shelf"(COTS) software that contained standardized interfaces for the import and export of certain standard definitions.

By the end of 2000s, XML became a standardized approach in defining interchange formats in internet applications. Its factors like flexibility, low costs and internet economy allowed the SMEs to participate in automated B2B interactions. Additionally, traditional EDI formats with a delimiter based approach did not meet the current B2B state-of-the-art business requirements, which added one more reason for enterprises to consider markup-based languages like XML.

1.1.2 Changing Face of EDI

With the beginning of EDI, much attention was given to the format of data capture. In other words, for a long time e-business activities were conducted focusing on the standardization of message formats, document types and application specific descriptions for stable pairings of trading partners. However, e-business activities did not meet the current changes in business process development nor the gathering of a dynamic set

of business requirements required during collaboration. XML and EDIFACT are two leading examples.

With a major re-thinking of the subject in the B2B field, a paradigm shift from document centric to process centric activities on the business level has been observed. On the business level, business process models capture the business logic, independent of the technology platform. These models are subsequently implemented later on different technology platforms. Through this approach in hand, various enterprises can quickly adapt to the changing business environment since business technology does not change as fast as information technology.

The B2B field offers potential to both research and industry. Therefore, in both sectors, an overwhelming quantity of specifications and approaches were observed. In order to standardize the multitude of e-business solutions, UN/CEFACT envisioned the idea of UMM and UPCC. UMM is a standard taking a more process-centric approach. UMM considers how data will be used, by temporarily grouping several business partners and managing business processes, extended throughout the value chain. Moreover, this approach provides flexibility in terms of adaptability to new business requirements and syntax-independent modeling of business processes. In regard to the information exchanged between business partners in business processes, UMM supports the use of UPCC. UPCC uses the core component library, where users are restricted to model business data based on existing core components that are already defined.

Business processes are the core assets of any organization. They produce value and justify the company's existence. The best way to implement business processes is by aligning them to strategic goals of an organization. Once these goals are identified, it becomes easier to realize organizational goals. Usually, while working in a collaborative B2B environment it is evident that each participant describes their own internal business processes. As a result, shared business processes descriptions between business partners described from different perspectives do not align with one another. This leads to misinterpretation between business partners. An approach is required that captures business processes from a global perspective in order to obtain an efficient business process model. Business enterprises need to correspond with each other in areas of shared business activities and business information exchange [43].

Moving further, the implementation of shared business processes are based on verbal communication between business experts. This leads to a risk of misunderstanding, bilateral agreements between business partners and loss of information. There is also a second related problem during the implementation process. While the developers might be interested at the technical level and the stakeholders at a non-technical level, it is hard to communicate and overcome the different vocabulary between them while capturing the business requirements. Hence, a high level of abstraction is necessary, through which the model driven approach is used to bridge the gap between the domains of business and IT. Thirdly, while the concept of business might not change, the diversity

and constant evolution of technology can introduce new syntax elements modifying the semantics leading to multiple business document standards. This can include migration and maintenance costs which only large companies might afford. In order to avoid the above mentioned problems, a conceptual representation for inter-organizational business processes that later proposes a generic meta-model is necessary.

Two kinds of business process modeling approaches are presented to obtain dynamic B2B e-commerce: *top-down approach* and *bottom-up approach*. A top-down approach begins on a business level and then drills down to the operational level. In this approach, business processes are not conducted in isolation and are best suited for stable requirements. A bottom-up approach deals with business scenarios that define key activities to be performed in order to obtain business process goals. However, both approaches are reasonable and viable. Most experts agree, that in practice a pure application of these two approaches does not work. An iterative business process that uses both a top-down approach as well as a bottom-up approach is often the best as it allows the model to be tested from both perspectives [45].

A business process can be conducted within an organization or between organizations. A key characteristic of inter-organizational business processes is the fact that they involve multiple parties. These parties perform different parts of the overall process, whereas no one has control over the global state. This is similar to a multi-application system business processes whose parts are executed by different autonomous systems within one organization [11]. In an inter-organizational setting, different parties usually have to balance a need for coordination in order to realize the maximum efficiency with a need of privacy for their value-creating procedures. In order to deal with this requirement, a distinction is made between the interrelated views of inter-organizational processes: global choreography, local choreography and orchestration [12].

Choreography refers to message sequences exchanged between different parties in an inter-organizational business process [13]. Choreography languages cannot be directly executed and need to be mapped to an orchestration language in order to be executed. Local choreography describes the flow of information from a participating partner's point of view. It makes the public parts of its local process visible to its partners. A global choreography defines inter-organizational business processes from a neutral perspective [14]. Orchestration refers to an executable part of an inter-organizational processes provided by one party. This executable process or integration process [11] interacts with both external web services of other business partners and internal services. According to the definitions, choreographies are relevant for the specification of inter-organizational processes, whereas orchestrations are used for the composition of internal processes [14].

Due to of the essential role of documents in a business, it is quite natural that document analysis is a major part in e-business. Business documents are built on business data, captured during business interactions. Among several business data modeling ap-

proaches some of the modern approaches are entity-relational, relational, semantic and generic. While data modeling concentrates on a multitude of business documents, document modeling concentrates on reuse of modeling elements required for building a business document model.

Independent of individual perspective, there are conceptual diagrams which model the reality of business interactions. Conceptual modeling is the core of any business process modeling process which is capable of defining and describing the relationship between object concepts. UML, Object Role Modeling (ORM) and Entity Relation (ER) are the most popular choices and are used to describe documents at the conceptual level. Conceptual business document modeling is building the business document based on the building blocks (relationship, diagrams) along with its rules and general mechanisms (specifications, extension mechanisms). Initially, UML had been adopted as the standard modeling language for modeling software systems. However, with time it was extended to support business modeling.

1.1.3 B2B Open EDI Reference Model

As the idea of technology-neutral business process models became realistic in business markets, the next step of evolution was the idea of exchanging electronic data among organizations without prior agreement. A requirement towards a fast and dynamic way of conducting business was found necessary, resulting in an Open-EDI reference model standardized by the International Organization of Standardization (ISO). Open-EDI is an approach that enables organizations to establish short term relationships quickly and cost efficiently. It introduces standard business scenarios required to support electronic data exchange. Once a business scenario is agreed upon, there are no prior agreements required between business partners other than the decision to engage in an Open-EDI transaction in compliance with the business scenario. The Open-EDI model is characterized into the Business Operational View (BOV) and the Functional Service View (FSV) as shown in Figure 1.1. The semantics of business data in associated data interchanges, business information, business convention, agreements and rules of business transactions and collaborations are covered in BOV. The specifications within the BOV capture business knowledge in a technology independent way. The FSV deals with information technology aspects that addresses technologies, supporting services and implementation aspects to support business collaboration specified in terms of BOV related specifications [6]. Candidate platforms on the FSV are traditional electronic interchange platforms such as UN/EDIFACT and ANSI X12 as well as more recent technologies like Web Services and ebXML.

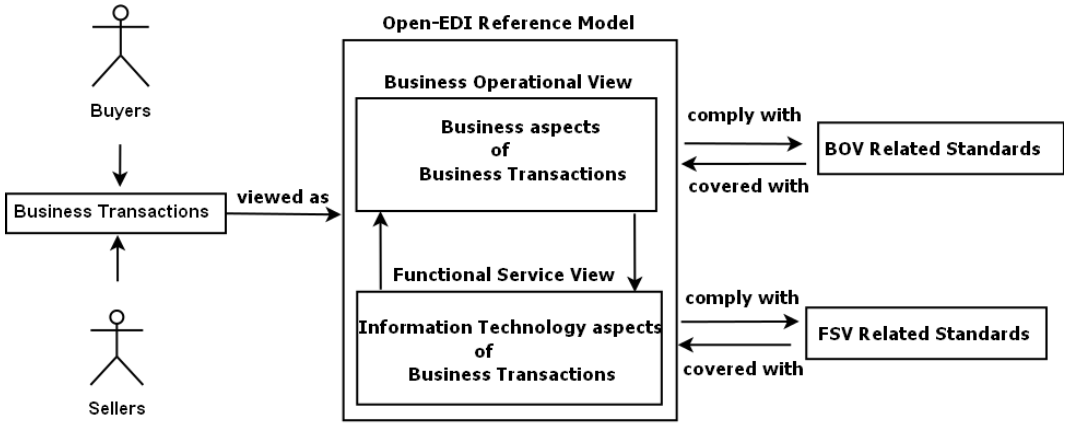

Figure 1.1: Open-EDI Reference Model

1.2 Contribution

Problem: In order to design business processes and information flow of business data between trading partners in UMM and UPCC, one needs to have a knowledge of both. Basic and sophisticated issues can initially be time consuming. Even though the UMM standard is well developed and documented, its complexity and strength make it difficult to initially understand for a novice user. For a beginner it might not be apparent when and what kind of artifacts are to be created and how efficiently they can be utilized. Modeling concepts such as core components, stereotypes, tagged values and constraints might be unfamiliar to a novice user. On the other hand, an experienced modeler can fall prone to errors such as overlooking the required strategy, scope and reuse of artifacts utilized during the design process leading to an inefficient model full of errors. Hence, a user guide for UMM and UPCC is necessary to assist any kind of modeler while creating a meta model.

Contribution: The contribution of this thesis is to provide a step-by-step user guide for UMM and UPCC based on two perspectives. The first perspective is based on an abstract point of view where the concepts are explained based on a neutral and platform independent way. The second is based on a modeling tool perspective with the help of an example model.

From the conceptual point of view, this user guide starts from simple concepts that any modeler already understands and incrementally adds new semantics and activity flow. Following a good start, advanced patterns are detailed until a valid business process model is defined. A foundation is laid by describing the purpose of core components, stereotypes, tagged values and constraints used while designing the meta model.

UML diagrams are used to describe the scope and the activity flow of artifacts created during the modeling process. Furthermore, a definite order through which business information is exchanged during business activities is also described here. In this guide, UMM uses UPCC as a business document modeling technique. Library descriptions and efficient reuse of core components are also detailed here. This user guide incorporates some parts of the UMM [8] and UPPC specification [74].

From a tool perspective, a stepwise approach is explained in the second part with the help of an example. The University of Vienna along with the Research Studio Austria have developed a plug-in for the UML tool Enterprise Architect known as VIENNA Add-In (Visualizing Inter ENterprise Network Architecture).

1.3 Web Page and Teaching Supplements

A web page for UMM and UPCC is located at this URL: `www.umm-dev.org`. This web page also contains:

- Examples models for UMM and UPCC
- UMM Specifications - Base and Foundation Module
- Case studies
- Teaching material
- Tools - Worksheets, UML profiles, VIENNA Add-In

This guide assumes that the user has a basic knowledge regarding the fundamental concepts of UML. In case of the user being new to UML, it would be an advantage to read an introduction to UML first, mainly from sources such as [9] or [10]. This user guide is mainly of interest to modelers who include architects, business managers, analysts, consultants, designers or business people taking part in business process modeling. This thesis serves as a complete reference guide for modelers from a blank slate to a complete valid UMM compliant model, with detailed information regarding sub packages, views, tagged values, constraints and stereotypes. All concepts of UMM 2.0 based on UML 2.1.2 are covered in this guide. Other versions of UML may not deliver the required results to the models. Furthermore, this user guide is described based on the Open Development Process Step 2 (ODP6) iteration profile version for modeling UML Profile for Core Components 3.0.

The concepts with intuitive descriptions, many of which are provided on the running example “Order from Quote” example model are also available at the following link: `http://umm-dev.org/example-models`.

CHAPTER 2

Related work

2.1 Business process modeling standards

Unified Modeling Language (UML) 2.x. UML a general purpose modeling language defines a set of graphical notations which assists the modeler in describing software systems. UML [50] has become an accepted standards applied in business, from requirement gathering to implementation of processes or systems. UML 2.x provides a rich set of behavioural models which are very useful in modeling processes, activities and information, critical to every business. UML activity diagrams, one of its artifacts are typically used for business modeling choreography. Two major categories are identified in UML2 - UML infrastructure and UML super structure. While the UML infrastructure is too abstract to model software applications, the UML super structure defines the concepts that developers use to build UML models. One of the main dimensions proposed in UML2 is the Object Constraint Language (OCL) that specifies queries and invariants in UML models. While the basic concepts of diagrams remain the same, in UML2 additional features and enhancement are observed in the interaction diagrams. One such example would be the interaction overview diagram where the activity diagram serves as a "master chart" for a set of sequence diagrams. Significant changes and incremental improvements are also noticed in architectural modeling such as structured classes where a class is composed of parts with an explicit "nested" notation.

Beyond the standard UML notation, a UML "extension" enhances the capturing of business processes and its constructs, the most important which are Eriksson-Penker Business Extensions [54] and Rational UML Profile for Business Modeling [54]. The Eriksson-Penker Business Extensions merge UML with process modelling, providing a much needed specialised UML extension able to handle business process modelling. Using these extensions, the business process modelers may add stereotypes and/or properties to the UML in order to suit their particular situation.

Usually document semantics are defined at the logical level which are syntax dependent. Extensible Markup Language (XML) schema is a widely accepted representation format for storing and exchanging structured and semi-structured information at the logical level. Today, with an increased use of XML standards that provides secure and reliable exchange of business data with inexpensive networking standards, a lot of effort on research work is put upon conceptual-level UML to logical-level XML (forward engineering) and XML to UML (reverse engineering) mapping processes. In forward engineering, a conceptual diagram is designed such as UML, from which an XML schema representation is automatically derived. Various methods are presented where UML conceptual data models are transformed into logical XML syntax such as UXS (UML and XML) [17] and uml2xsd [18]. On the other hand, in reverse engineering, conceptual models such as UML are generated from XML schema artifacts. A reverse engineering solution from XML schemas to conceptual diagrams is proposed by Necaský in [19]. Even though these concepts have been addressed, they have not yet been successfully applicable in practice, since not all features of XML that reflect from different perspectives can be represented as UML by default. In [16], a XML conceptual model - XUML, based on UML 2.0 unveils storing data centric documents exchanged between enterprises.

Business Process Modeling Notation (BPMN) 2.0. The latest version of BPMN released by the Object Management Group (OMG) early 2011 [30] is a widespread standard, graphical modeling notation used to model business processes. Its primary goal is to provide a notation that is easily understandable by all business users, from business analysts who create initial drafts for business processes to technical developers who implement the technology and finally to business people who will manage and monitor those processes. However, this notation is highly recommended at the business process design stage for modeling business choreography.

Among other improvements, the choreography diagram is one feature introduced in BPMN 2.0. In the previous versions of BPMN, the only way to represent choreographies was via collaboration diagrams. In the new version, choreography diagrams show the interaction between participants in a different format by concentrating on the message flow than on the individual detailed task of business processes. The atomic building block in choreography diagrams is a choreography task. An atomic task represents an interaction, which is one or two message exchanges between two participants. Together with the choreography tasks, come message flows relating the interaction with an initiating message (represented by a white envelope) and, possibly, a return message (represented by a black envelope). This yields one-way interactions or two-way interactions. A refined choreography with several interaction are contained in a Sub-Choreography. BPMN 2.0 introduces the concept of Call Choreography, a wrapper for a globally defined choreography and the plus symbol denoted a call to a Sub-Choreography. Mul-

tiplicity can by indicated by three black parallel lines in the bottom center of the pool which represent participants. In order to describe the control flows, sequence flows are used for performing two tasks in a sequence or gateways are used for more complex behaviours. Other BPMN 2.0 element like activities, events, timers, exception and compensation are used to supplement the choreography descriptions.

Decker et al. [58] exhibits ten requirements put on choreography languages. An evaluation in [56] using these requirements show whether BPMN is suitable for choreography modeling. A requirement that is not supported by BPMN choreographies is *Message Multiplicity* which captures the definition of multiple messages sent from one or more participants to others. This shortcome avoids fulfilling the so-called multi-transmission interaction patterns. Another detected problem is the weak support for *Reference Passing* where participant A enables participant C to communicate with participant B by passing the reference of B to C. This avoids fulfilling so-called routing patterns. A larger need is observed in the meta-model quality where the information is presented in a very technical level. Based on the evaluation carried by the authors in [57] regarding BPMN 2.0's constructs, more shortcomings, challenges and open questions of BPMN 2.0 choreography diagrams are discussed in [55].

Domain Specific Language (DSL). Instead of modeling business processes or documents that tailor a general purpose language, taking for example, UML through Model-Driven Development (MDD) [20], one possible solution would be to define a DSL that lowers the ambiguities on conceptual models for a specific environment. Unlike UML and BPMN, DSL's provide means for concise representation of semantics of the particular business domain, enabling the development of consistent and expressive business process models. Resulting models can be used not only as specifications for information systems, but also easily accessed by programming languages, creation of documentation, testing and other purposes through which the main goal of Model-Driven Architecture (MDA) is achieved. In order to describe a certain domain, specific vocabulary on this part of the sphere is applied. Hence, not just the resulting model but rather the modeling language itself has a semantic connection with the application domain. In [53], a case study shows how a domain specific language called ProMod, an extended subset of BPMN is used to model business processes for a domain. Not only is a DSL used in business process modelling but proves its mightiness in business document modeling as well. [21] guides a business document modeler, towards creating a valid core component modeling environment based on a DSL.

DSL offers a couple of advantages. Business domain experts are able to understand, modify and validate within the language. Due to its concise nature, DSL's are self-documenting up to a certain extent. A DSL has not only advantages but also potential shortcomings. One major drawback is the knowledge required in design and language, a DSL developer needs to have about the target domain. Other problems may

include performance, tooling and user training costs.

Electronic Business using eXtensible Markup Language (ebXML). ebXML, a successor of EDI was started under the initiative of UN/CEFACT and OASIS. The main goal of ebXML is to create an electronic marketplace, where companies can find each other, agree to become trading partners and conduct business. This way, a standardized means to formalize business processes is provided thereby ensuring interoperability between ebXML compliant systems. Business activities are performed automatically without any human intervention over the internet.

A basic part of the ebXML infrastructure is the *Repository/Registry*. It stores important information about business along with the products and services they offer. The trading partners use these services to register their business profiles and partner's profiles by creating *Collaboration Protocol Profiles (CPP's)*. To model the collaboration in which companies engage, ebXML defines *Collaboration Protocol Agreements (CPA's)*. A *CPA* is an agreement which specifies the conditions under which the trading partners will collaborate. ebXML does not mention that any modeling methodology is mandatory. If a designer decides to use one, then UMM is a methodology of choice used to create business process models. In ebXML, business document standards are unambiguously defined by *core components*. They are building blocks of XML semantics and business vocabulary used in messages and documents. The purpose of core components is due to the fact that is is possible to retrieve and store core components easily in the standard ebXML registry. In [28], guidelines for making strategic decisions concerning ebXML implementations are featured. Further on, the author explains how major B2B vendors plan to integrate ebXML into their products.

Some of the major drawbacks of ebXML are that the specification is not entirely complete and the support of industry is still lacking. It is not so easy to handle due to the fact that it is powerful and its implementations are likely to be complex. Templates for the common demand of companies might help decrease the time-to-market for system providers that use ebXML implementations [27]. Based on the case study of [29] in 2008, ebXML has failed to reach its goal due to the fact that it has failed to adapt itself in the digital world where the talk is about service architectures such as SOA and Web 2.0. The core component standard has also not meet expectations. Instead of being based on its business usage, the information structure is based on harmonized semantic structures that are not only different from normalized data but also not anywhere close to lexical structures.

2.2 Business document modeling standards

Core Component Technical Specification (CCTS) [31] is an implementation neutral standardized approach towards modeling business document information, developed by

the UN/CEFACT. It represents common types of business data that are presently used in the industry. This specification describes in detail, the technique through which syntax-neutral, technology-independent core components are discovered and created for assembling business documents. It ensures the possibility of restructuring and creating new business documents to achieve interoperability of data. CCTS is put into practice whenever business information is shared between business partners from different departments or organizations. It is dependent on the UML, based on the way how it is expressed but does not require UML in its implementation. It focuses mainly on two concepts: *core components* and *business information entities*.

Core components are components which appear in different sectors of business. When the core components are applied on a specific business context they are referred as Business Information Entities. The semantic definitions of core components as well as business information entities are defined in a so called Core Component Library (CCL) [35]. The CCL serves as a package that contains core components and is identified in the ebXML compliant registry making its contents available to the core component user community. A business modeler may then create his own business specific content based on these core components. Some of the major benefits of core components include re-usability of artifacts created, interoperability across organisations and consistency in standards. On a contrary, a major drawback of core components is the restricted number of core components from predefined sets in the CCL, due to which the modelers are not able to create user defined core components. Other examples for implementation neutral standard include the Context Inspired Component Architecture (CICA) [32] maintained by Electronic Data Interchange (EDI) where business document messages can be expressed as an XML schemata.

CCTS follows the Open-EDI reference model concept that separates the business aspects from technical implementations; *refer Section 1.1.3*. A user guide for CCTS is provided in [33]. It is divided mainly into three parts. The first and the third part are the ones which are of most relevance for CCTS modellers. The first part is focused on the implementation of CCTS 3.0 and the third part is focused on UML models such as UPCC 3.0 and XML for CCTS.

Universal Business Language 2.0 (UBL) [22] is an OASIS standard in the form of an XML syntax, that provides a set of reusable data component such as Item, Price, Address for a set of document schemas such as Invoice, Order, Statement, Quotation. This way UBL is the first specification that implements the CCTS of UN/CEFACT, providing a foundation of predefined reusable elements. UBL defines business documents in business areas of sourcing, ordering, invoice and fulfilment. In other words, UBL is a library of BIE's based on CCTS. Due to its generic nature, it provides the possibility to confine, extend, modify or create ones own business documents.

UBL defines a set of business definitions that utilize the core components of CCTS.

It uses the concept of ABIE, BBIE and ASBIE to build document schemas: *refer Chapter 6, Section 6.1*. In UBL 2.0 naming and design rules, BBIE and ASBIE of CCTS are realized through *BBIE property* and *ASBIE property*. The CDT's of CCTS are imported in UBL and are termed as *Unqualified Data Types*. There are also *Qualified Data Types* which are used as code lists in UBL. The data type of a *BBIE property* is specified through an *Unqualified Data Type* or a *Qualified Data Type*. Additionally, UBL 2.0 introduces an optional element *UBLExtensions* in documents, which is used in to include non-UBL data elements thereby accommodating the widest possible range of extensions.

Business documents in UBL 2.0 are validated through two-phases. In the first phase UBL documents are validated against a standard UBL 2.0 XSD schema. If the first-phase validation is passed, the second phase checks against the rules specified through XSL or schematron languages which specify additional constraints on the code list values of the elements in the instance. UBL can be customized using two techniques namely *conformant* and *compatible* customization. In a conformant customization, an instance is validated against a standard UBL 2.0 schema. There are four ways of perform conformant customizations. Firstly, by utilizing the *UBLExtensions* element. Secondly, by subsetting the original UBL 2.0 schemas based on the requirements. Thirdly, by customizing the code lists. Fourthly, by applying additional constraints on the value space of information entities instance. On the other hand, a compatible customization performs more complex modifications such as modifying the BIE artifacts.

In [59], one of the shortcoming pointed out regarding UBL is the fact that the standard, in general, has a quite overloaded structure, where all possible requirements are captured, similar to top-down standards. Based on a collaboration agreement between OASIS and UN/CEFACT, UN/CEFACT recognises UBL 2.0 as an appropriate first generation XML documents for e-business. The UBL library is merged with the UN/CEFACT's Core Component Library, since it is a useful aid to interoperability between vocabularies. Even though the maintenance of UBL remains with the OASIS, it grants UN/CEFACT, a perpetual, irrevocable licence for creating derivative works based on UBL [60].

United Nations/Electronic Data Interchange For Administration, Commerce and Transport (UN/EDIFACT) is a formal messaging language within the EDI approach used in B2B transactions that include machine-readable descriptions of electronic business documents. It is developed and maintained by the UN/CEFACT [4]. UN/EDIFACT defines business documents in a standard format and are constructed based on a set of syntax rules. Having standard formats means that all trading partners understand the document structure and interpret it correctly.

An EDIFACT message contains many sections called segments. Each segment contains many sub-sections called composites, and each composite contains none or more

sections of information called data elements. The EDIFACT standard defines what segments are allowed in which types of messages. Data elements within composites are delimited by semicolons. It should be mentioned that there are special circumstances when a delimiter will need to be interpreted as normal character and not as a delimiter. In that case, it's detectable by observing a question mark in front of the delimiter. Figure 2.1 shows an example of an International multi modal status report message (IFTSTA) used in EDI to report the transport status between trading partners.

```
UNA:+,? '
UNB+UNOA:2+CARRIER+LOSE+050405:1503+36'
UNH+1+IFTSTA:D:01B:UN'
BGM+7+523374+9'
DTM+137:201208051330:203'
CNI+1+000001661753'
STS+1+54+231'
DTM+9:201208090647:203'
CNI+1+000001661754'
STS+1+21'
DTM+9:20120809:102'
UNT+10+1'
UNZ+1+36'
```

Figure 2.1: EDIFACT Message Format

The EDIFACT syntax is specified by the International Standard Organisation (ISO) 9735 standard. UN/EDIFACT messages are structured according to very strict rules. The message formats covers mainly three areas namely administration, commerce and transport. Though this standard is used since almost four decades, it is still dominant in industry. However, EDIFACT is also stated as complex and requires a considerable amount of effort in realising business service interfaces, typically responsible for validation of sequences, grammar and syntax of incoming business documents which usually only large companies can afford.

Unlike markup language documents, UN/EDIFACT interchanges may contain messages full of codes which makes it hard to read in raw format. Best features of EDI with XML or XML with EDI, based on the perspective are build on XML/EDI [61]. XML/EDI builds on the EDIFACT foundations in terms of semantic contents (message types, segments and data elements) and related UN code lists. In [46], the shortcomings of business document standardizations are discussed with regard to UN/EDIFACT and XML based solutions.

2.3 Other Approaches

Enterprise Architecture (EA): Different from business modeling and having a broader scope, EA is concerned with the whole enterprise which captures all business processes, applications, data and infrastructure of an enterprise. Enterprise architects, who are the backbone of an EA are responsible for identifying cross-process capabilities. Sparx Systems Enterprise Architect, a UML modeling and design tool, assists a modeler not only while modeling business processes but also in designing and construction of industry based applications.

The Zachman Framework [23] from EA is a widely used approach that can provide many stakeholders with a common picture about the business processes the enterprise will use and the technology that will be adopted to support the business process. Most EA modeling tools exercise, modifying business process models. Instead of creating new business process models, one can reuse the existing business process models which are also created and used for other business process models. Most commonly known enterprise architecture frameworks include The Open Group Architecture Framework (TOGAF) [24], Integrated Architecture Framework (IAF) [25] & Information Framework (IFW) [26].

CHAPTER 3

UN/CEFACT Standards for modeling e-business transactions

3.1 About UN/CEFACT

The United Nations/Centre for Trade Facilitation and Electronic Business (UN/CEFACT) is one of the main branch of UN Economic Commission for Europe (UN/ECE) and is an internationally recognized standard organization for improving the ability of business, trade and administrative organizations. There are five permanent working groups within the UN/CEFACT that serve distinctive purposes.

- The *Techniques and Methodologies Group (TMG)* [34] working group examines and provides information and methodologies concentrating on business processes and Information and Communication Technologies (ICT) specifications.
- The *Applied Technologies Group (ATG)* concentrates on EDIFACT syntax structures and XML syntax where the former focuses on EDIFACT related works and the latter on document structures based on specific technology that supports UN/CEFACT project work.
- The *International Trade and Business Process (TBG)* is responsible for capturing business requirements and initiating developments in areas of process analysis, best practices and international trade procedure where UN/CEFACT's Modeling Methodology (UMM) is applied when relevant.
- The *Information Content Management (ICG)* ensures that all the technical specifications of UN/CEFACT are released corresponding to the approved procedures, and are at the highest quality status.

- The *Legal Group (LG)* ensures that all the legal facets of e-business and international trade facilitation are acknowledged by the UN/CEFACT.

Taking the first working group - TMG into consideration, there are three subgroups defined. The *Business Process Working Group* (BPWG) is responsible for UN/CEFACT's Modeling Methodology (UMM) and Resource Event Agent Specification (REA). The *Core Component Working Group* (CCWG) takes in charge of Core Components Technical Specification (CCTS), Core Components Message Assembly (CCMA), UML Profile for Core Components (UPCC) and Unified Context Methodology (UCM). Finally, the *Electronic Business Architecture Working Group* (EBAWG) is superseded by the work of UN/CEFACT Standard Development Advisory Team (CSDAT). Overall, TMG is responsible for the following modules:

- *UMM*: An inter-organizational business process modeling methodology based on UML.
- *REA*: An enterprise ontology whose specification module is to provide specific guidance to UMM users for developing UML diagrams necessary for executing business activities from the business requirements view; *refer Chapter 5, Subsection 5.1.1* of UMM [49].
- *CCTS and Core Component Library (CCL)*: While the former is developed to model business information independent of syntax specific platform and to ensure interoperability between organizations on the data level, the latter represents a repository that stores the core components needed for UMM implementation.
- *CCMA*: A special methodology for assembling higher level Business Information Entities (BIE's) for electronic messages.
- *UPCC*: An unique representation mechanism for core components in UML based on CCTS. It is used to represent business documents exchanged during inter-organizational business processes.
- *UCM*: It provides a mechanism for developing, registering and using context drivers as part and for the application of UN/CEFACT standard artifacts.

3.2 Overview about UMM

In the early days, EDI focused on a document-centric approach that created common business document standards during business interactions, in order to avoid bi-lateral agreements. With the evolution of internet, a shift is noticed from document-centric approach to process-centric and service-oriented one. UMM is one such process-centric

approach that focuses on business activities than on common business document standards.

UMM, a modeling methodology notated in UML syntax, captures business requirements. Capturing this business knowledge enables development of low cost software components useful for small and medium sized companies. Ultimately, its main aim is to identify business processes, develop a global choreography of inter-organizational business processes and their information exchanges. Developing business processes in a technology-neutral manner provides an insurance against technical obsolescence.

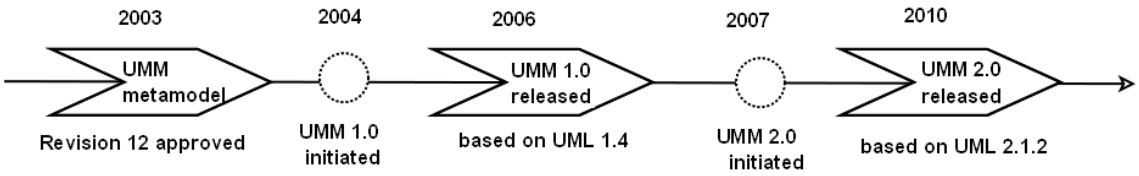

Figure 3.1: Evolution of UMM

The UN/CEFACT TMG approved UMM revision 12 after the completion of review process in 2003. UMM 1.0 was first initiated in the year 2004. Before long, UML 1.4 was the current version with UML 2.0 under development. UML 1.4 was taken into consideration by the UN/CEFACT group. Eventually, a UML foundation module [7] was released in 2006 that became the technical specification for UMM 1.0 [66] on top of UML 1.4. Since 2007, UML 1.4 seemed to be outdated. Various modeling tools started using UML 2.0, due to changes observed in the notation of diagrams, structural modeling and further developments in UML profiling. As a result, it was found necessary to migrate to UMM 2.0. In 2007, a whole new modeling structure of UMM 2.0 was initiated by UN/CEFACT and in 2011, a foundation module technical specification [8] for UMM 2.0 built on top of UML 2.1.2 was released. Figure 3.1 shows the progress of UMM.

UMM methodology proposes a three “step-by-step” top-down approach that creates a UMM complaint business process model. It begins with the capture of business knowledge from business experts followed by building business collaboration models, also resulting in designing business documents required during business collaborations. In certain cases, a bottom-up approach can also be used as a starting point which begins with updating existing business documents and transactions with new or modified business requirements. Through this approach, existing artifacts are used to create business process models.

3.2.1 UMM Modules

UMM meta model is classified into a set of modules, each serving different purposes ranging from simple to complex functionalities. The following partitions [48] as shown in Figure 3.2 have been defined in the UMM meta model:

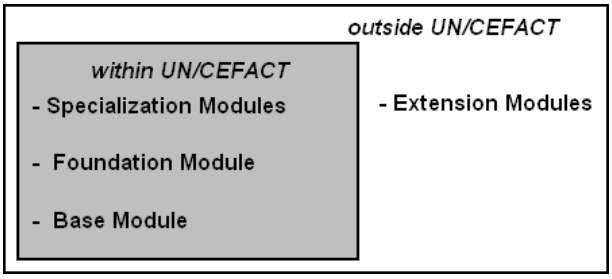

Figure 3.2: UMM Meta Model

- ***Base Module***: The fundamental principles used across all other modules are gathered [62].
- ***Foundation Module***: The core concepts required define a UMM compliant model and fundamental principles used across all other modules are elaborated and designed in the foundation module [8].
- ***Specialization Module***: Several specialization modules might define add-on concepts to the foundation module which are created and developed by UN/CEFACT. Each of a specialization module introduces a specialized type of analysis that extends the foundation module at a well-defined extension point for a specific type. Specialization modules may become candidates in the involvement of the foundation module.
- ***Extension Module***: They are additional features created and maintained by organization(s), external to UN/CEFACT and serve the same purpose as that of the specialization modules.

Following stereotypes are defined in the UML base module and used in the UMM foundation module or in UMM specialization/extension modules:

- ***Business Library***: A package stereotyped as *BusinessLibrary (bLibrary)* contains details of business processes and their related information. It is a container of elements that need to be registered in the registry and retrieved together, to be semantically complete [62].

- ***Business Information***: An abstract class stereotyped as *BusinessInformation (bInformation)* representing all types of exchanged information during business communication between trading partners. Since *bInformation* does not specify the type of exchanged business information , the concept of *Information Envelope* is used.

- ***Information Envelope***: A class object stereotyped as *InformationEnvelope (InfEnvelope)* within the *business information view refer Chapter 5, Subsection 5.1.3* of the UMM model. The *InfEnvelope* is a specialization of *bInformation* representing a specific type of exchanged business message during a *business transaction*.

The *business library* contains the following tagged values:

- *uniqueidentifier*: A unique identifier represents the *business library* package in the registry. A Universally Unique Identifier (UUID) is used within the URI structure scheme to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library.*
- *versionidentifier*: The version identifier represents the current version of the *business library* in the registry. The version information should not be assigned by the modeler rather must be managed in the registry. An unregistered library must not have any version assigned.
- *businessTerm*: A synonym frequently used for the stereotype in business.
- *copyright*: Copyright of the *business library* package.
- *owner*: The owner of the *business library* which could be an organization, institution or an individual.
- *reference*: It contains location information to additional resources where further information about the *business library* could be found.
- *status*: It represents the current lifecycle status of the registered *business library* package. The status is set by the registry. The possible status representing the *business library* are *proposed, approved, mandatory, validated* and *implemented*. An registry unregistered must not have any status information assigned.

3.3 Overview about UPCC

When ebXML was originated, one of the key issue was an unambiguous definition of capture and reuse of business information between trading partners that used different syntaxes (XML, UN/EDIFACT) with a semantic standardization. A solution was to

define a common base in the ebXML framework. With an evolution of time, a powerful way of defining interoperable business documents in a syntax-neutral manner was introduced through a concept known as *core component*.

Core components, a general approach towards defining business documents are standardized through regular spread sheets. In order to put these core components into industrial use, a representation of choice needs to be determined. Due to its implementation neutral standard definitions and generic nature, core components compliant business document models may be built with any appropriate representation. The core components follow a top-down business document standard approach, which is one of the predominant standard paradigms.

The core component concept, later taken over by UN/CEFACT was further developed and in the year 2002, an extended standard was initiated, known as the Core Component Technical Specification (CCTS). CCTS defines a meta model for core components in a non-formal manner that assists the modeler to identify, capture and maximize the re-use of business data to increase interoperability between business processes. In recent years, UML modeling language has been increasingly used in the fields of business information modeling and is supported by a broad variety of modeling tools. CCTS which is also described using UML however, does not require UML in its implementation. The latest specification of CCTS version 3.0 was released in the year 2009 [31].

CCTS defines its own MOF meta model which is entirely independent of the UML meta model. Therefore, no unique representation mechanism for core components in UML is provided. If every modeler were to define their own UML representation mechanism then the resulting core component artifacts are unlikely to match during collaboration. Additionally, the storage and retrieval of core component artifacts in a centrally accessible registry is impossible, since no commonly agreed representation format of core components is available. This is a strong contradiction of the initial purpose of core components. When UML is chosen as a language for modeling core components, the core component meta model is described using the UML Profile for Core Components (UPCC). UPCC, used for specific needs of business document modeling, tailors the UML meta model based on the domain requirements by using stereotypes, tagged values and constraints. An overview of UPCC is shown in Figure 3.3.

The first version of UPCC 1.0, based on CCTS 2.01 was released in the year 2008. The latest version of UPCC 3.0 is based on CCTS 3.0. CCTS 3.0 distinguishes between *basic core components* and *basic core component properties* as well as between *association core component properties* and *association core components (ASCC)*. However, in UPCC, only the concept of *basic core components* and *association core components* are used.

UPCC provides conceptual business document models based on UML. In order to use these conceptual business documents for a business data exchange, a logical level representation of data is required, such as XML schema. XML schema files can be

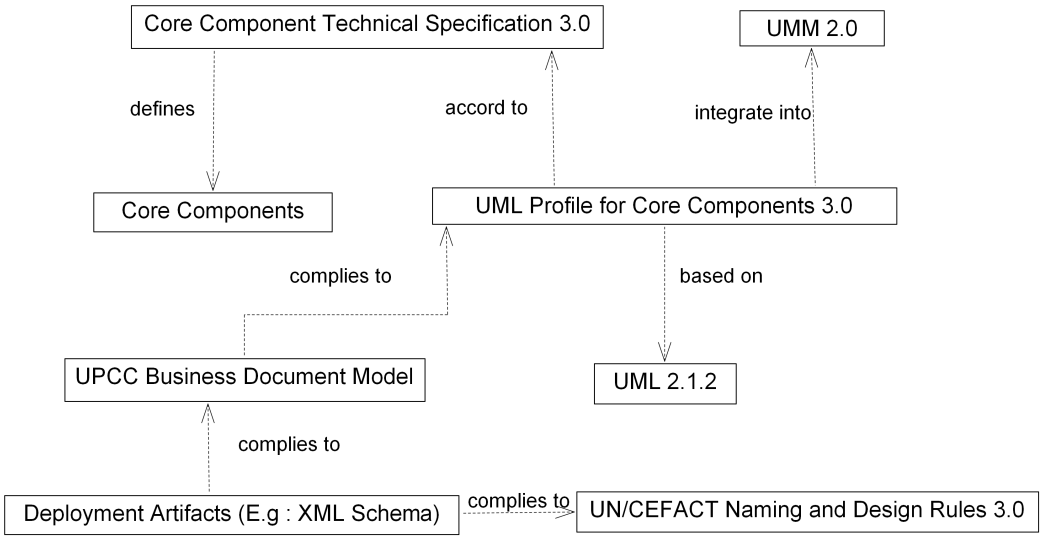

Figure 3.3: Overview of UPCC

used to validate exchanged business documents in B2B collaboration. To define a XML schema, it is necessary that the conceptual model follows a set of restrictions. Stereotypes, tagged values and constraints defined in the UPCC are used to implement the validation routines to ensure the model validity against the core component standards. These validated models can then be used as a foundation for the derivation of deployment artifacts. If UPCC were not used and the modeler were to define his own UML representation format then the core components wouldn't match and the unique representation of XML schema would likely fail. In CCTS, the rules for core components can be specified in any constraint language of choice. However, in UPCC, the Object Constraint Language (OCL) is the only constraint language that defines rules where *invariant* is the only condition used during modeling.

In order to store and retrieve the conceptual as well as logical layer of business information, a registry meta model based on ebXML registry information model [38] is used. The UML syntax of the core components, represented as XML Metadata Interchange (XMI) are stored as extrinsic objects within the registry. Since the objects within the registry are encapsulated they cannot be retrieved directly from the registry. Hence, they need to be annotated with relevant metadata to yield an adequate reply. The registry defines required dependencies between objects of different artifacts and metadata if needed. Currently, the Information Content Management Group (ICG registry) of UN/CEFACT is developing a UN/CEFACT Core Component registry implementation specification [39].

The purpose of using UPCC in business document modeling is mainly due to four

reasons. Firstly, the precise and unambiguous definitions of core components in UML through UPCC assists the business modelers customize their tools to be CCTS complaint, thereby minimizing the complexity to the business modeler. Hence, UPCC can be used by any UML modeling tool of choice, providing the business modeler the necessary artifacts to define business documents. Secondly, the issues of interoperability are also resolved overcoming several standardization initiatives. The UPPC provides a mapping from the CCTS concepts to its corresponding UML elements as a UML profile. Thirdly, this conceptual model built in UML is integrated in SOA context by representing in a logical level i.e. XML Schema. Finally, the integration of UPPC in UMM enables the modeler to design the business process models and business document consequently [37].

3.4 Relation between UMM and UPCC

UPCC along with UMM is an integrated approach towards capturing the collaborative space between organizations from a business domain. UMM models the choreography of business data exchange commitments agreed between trading partners. It provides three views for structuring business process modeling activities. The *Business Requirements View* gathers as many business requirements as possible from business partners, organizations and individuals needed for modeling inter-organizational collaborative business processes. The *Business Choreography View* defines the steps required for exchanging business information during the business process collaboration. Finally, the *Business Information View* serves the purpose of defining artifacts for business documents, modeled for the information exchange.

The *Business Information View* combines both UMM and UPCC . *Core components* which are not a part of UMM, are integrated into the *Business Information View* through UPCC - a meta model, that is tailored to a specific domain, using the concept of *Business Information Entities*.

3.5 UML Profile

UML language elements such as classes, attributes, associations, packages, collaboration etc. are described and extended through an UML metamodel. A metamodel is constructed from a set of Meta-Object Facility (MOF) classifier types proposed by the Object Management Group (OMG) [15]. The MOF framework tries in a simple way to separate different levels of abstraction, based on a 4-level architecture. The metamodel which at the M2 level of the MOF architecture is used to describe the model of a model as shown in Figure 3.4 . The metamodel of UML can be extended via UML profiles for modeling domain specific concepts.

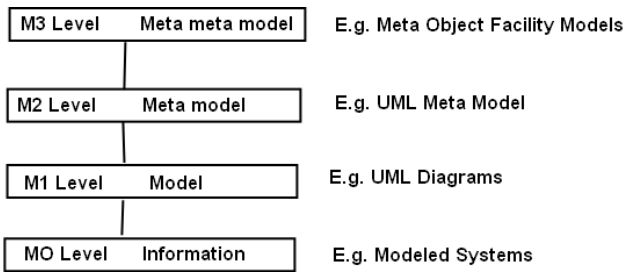

Figure 3.4: Meta-Object Facility Architecture

UML profile, customized for a domain and platform specific purpose, defines a coherent set of stereotypes, tagged values and constraints. This profile is created by specializing the semantics of standard UML metamodel elements. UMM is a UML profile used for the purpose of modeling inter-organizational business collaborations. UMM applies stereotypes for modeling elements to identify their purpose and extend their meaning. Tagged values define the properties of these stereotypes by specifying a key-value pair. Constraints identify the restrictions, pre- and post-conditions that require a modeling element to conform during modeling.

3.5.1 Stereotypes

Stereotypes are the core extension mechanisms of UML. A stereotype is a model element that classifies an UML element. A stereotype defines how an existing metaclass can be extended. Metaclass can be one of the elements of UML (e.g. class, interface, package, association etc.). This implies that an instance stereotype is used to associate with the instance of its related metaclass [36]. It refines the meaning of a modeling element and cannot exist without the extension of a modeling element. Stereotypes are used to describe a model element and are represented as text between two surrounding characters "≪ ≫" called guillemets. Furthermore, stereotypes have properties defined, in so called tagged values and pre-, post-conditions defined in so called constraints.

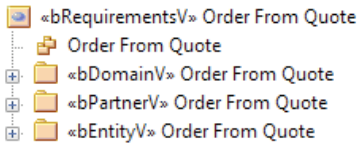

Figure 3.5: Stereotype Representation

In UMM, stereotypes are used to represent artifacts such as a packages, elements or diagrams. Figure 3.5 shows an element, a package diagram and three packages, all having the same name *Order from Quote*. Without the use of stereotypes all these elements wouldn't provide much information to the modeler. With the help of stereotypes *≪bRequirementsV≫, ≪bDomainV≫, ≪bPartnerV≫ and ≪bEntityV≫* these modeling elements are differentiated to serve different purposes. The package diagram is not assigned to any specific kind of stereotype. Furthermore, these stereotypes have different properties in their tagged values and different conditions defined in its constraints thus characterizing themselves from one another. Additionally, most of the important stereotypes are represented as worksheets; *refer Chapter 5, Section 5.2*.

3.5.2 Tagged Values

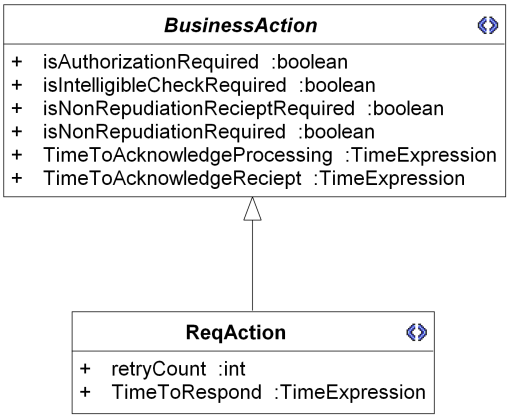

Figure 3.6: Representation of Tagged Values in a Class diagram

Every modeling element has its own characteristic or property, be it a stereotype, association, dependency or a package. The properties of a modeled element are added in so called tagged values. A tagged value is a key-value pair which gives additional information to an element. Tagged values are also considered as stereotype attributes. It is important to know that tagged values is not equivalent to UML class attributes. Unlike attributes, tagged values are defined not only to class elements but also to other modeling elements such as association, dependency, packages etc.

Figure 3.6 shows the representation of tagged values in a class diagram for an abstract stereotype *BusinessAction* and its child stereotype *ReqAction*. The stereotype *ReqAction* derived from *BusinessAction* inherits all its tagged values. Additionally, *ReqAction* defines two new stereotypes: *retryCount* and *TimeToRespond*. Figure 3.7 shows the initialization of the same tagged values for the stereotype *ReqAction*-`Obtain Quote`.

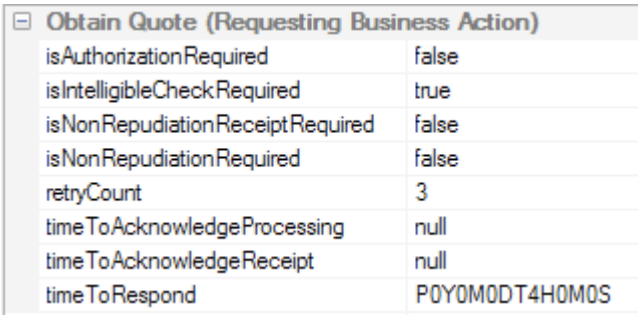

Figure 3.7: Tagged Values: key-value pair

During the modeling process, most of the tagged values defined for stereotypes are captured in worksheets; *refer Section 5.2*. Tagged values enable the modeler in specifying the properties of the business environment during business collaborations.

3.5.3 Constraints

Any UML diagram does not usually specify all relevant aspects of a specification. There are conditions or restrictions that exist on the degree of freedom of a modeling element. These cannot be expressed graphically and are normally documented as plain text called constraints. Practice has shown that plain text mostly leads to ambiguities. OCL is a preferred language in UML that is used to express constraints and written in the form of an expression that simply returns a value. *Invariants* help enhance the correctness of the model. An OCL Constraint can either be an invariant or pre- and post- condition.

Since UMM is based on UML, OCL is used as a constraint language in UMM. If all the OCL constraints rules within the UMM model are not violated then the model is said to be valid model. OCL constrains are described in the form of plain text language in this user guide and are not introduced on the technical level. Detailed information regarding OCL constraints on business collaborations can be found in [44].

Figure 3.8 shows an activity diagram of a *business transaction* pattern in UMM between two *business partners* along with their business activities. The restrictions on these business activities are not visible in this diagram. By defining a set of rules on these business activities a specific kind of *business transaction* is described. From the UMM foundation module specification [8], constraint 46 defined for this *business transaction* pattern as plain text is as follows:

CONSTRAINT: The BusinessTransactionPartition containing the RequestingBusinessAction must contain two or more FinalStates. Each of the FinalStates may have a SharedBusinessEntityState as predecessor. One of the FinalStates should reflect a Con-

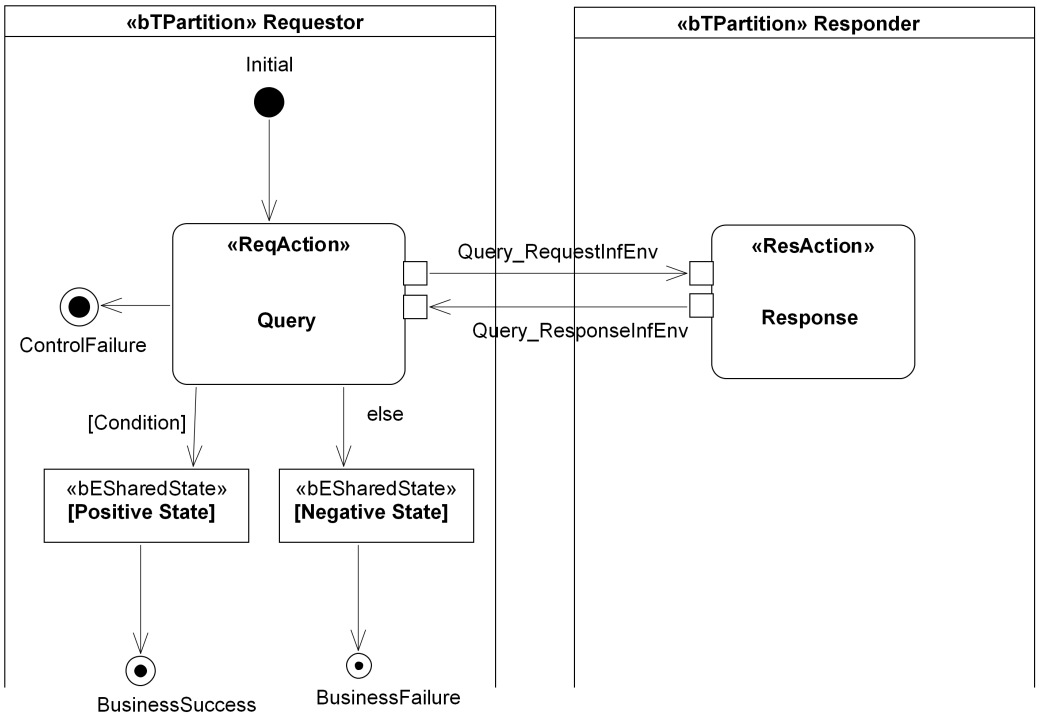

Figure 3.8: Business Transaction Example

trolFailure – this FinalState should not have a predeceasing SharedBusinessEntityState

The same constraint described in the form of an OCL which is an invariant is shown below. The unambiguous way of modeling rules are described in the next few lines of the OCL.

```
Context ActivityPartiton inv :  self.isBTPartition() and
self.containedNode ->
exists(action |action.oclAsType(Action).isReqAction())
implies self.containedNode ->select( finNode |
finNode.oclIsTypeOf(ActivityFinalNode)) ->size() >=2
```

CHAPTER 4

Exploring the Order-From-Quote example

A majority of the organizations, do not have a clearly defined process flow that articulates various groups that impact the quotes and orders. Order-From-Quote process is the least automated process of any enterprise. The systematisation and automation are different in different ways. Companies with large investment, manage their ordering functions through an Enterprise Resource Planning (ERP) system. Others, through a Customer Relationship Management (CRM) system or create a quote tool or some form.

In this section, we set the stage by defining a minimal yet realistic quote order. This definition is given by an informal textual description that is accompanied by depiction in Figure 4.1. Due to its informality the definitions are necessarily ambiguous and incomplete. Starting from a customer interested in purchasing a quote, we may split the sales quotation business process into three steps:

- Register the customer
- Request for quote
- Place an order

From the business point of view the scenario is as follows: A customer initially needs to register at the registrar. If the customer is successfully registered then a customer id is provided. If not then the registration is rejected and the customer is informed. Once the user is successfully registered, he can continue the quote request process.

With a customer id that the user has obtained, a quote requestor requests for a quote. In order to decide whether to establish a quote or not, the seller requests the bank to check the buyers credit. After receiving the result of the credit check from the bank,

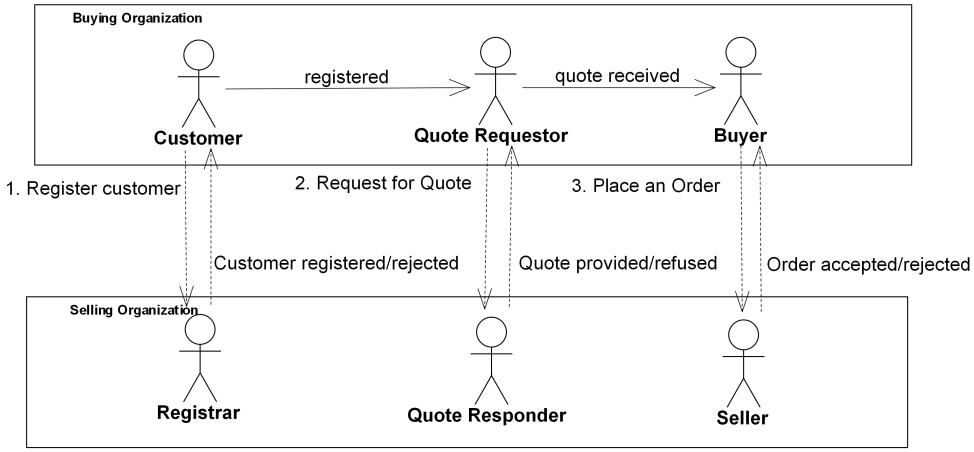

Figure 4.1: Workflow of the Order from Quote

the seller returns the quote documents, which either includes the quote or a reason for a rejection of quote. Most of the companies that deliver goods or services processes, generate and provide the requested quote to their prospects and/or create orders to fulfil a commercial agreement. In case, the quote requestor is found in a sanction list, then the quote is refused and the requestor is informed.

Once the quote is obtained from the quote responder, the customer reviews the quote. If the quote is acceptable, the quote is converted into a sales order. The buyer places an order and the order is submitted. If the seller accepts the order then the order is entered, processed and shipped to the buyer. If not then the order is rejected and the buyer is informed.

The example demonstrated, shows the basic workflow of a quotation process. Some of the following situations are not taken into consideration (i) when a quotation is not provided, the supplier may choose not to respond. (ii) if the quotation requested is incomplete, the supplier may request changes for the quote requested. (iii) the customer may ask for a revised quotation after already having received a quote. (iv) the workflow of an invoice.

CHAPTER 5

UMM User Guide

5.1 UMM Structure

UMM concentrates on business semantics, describing inter-organizational business processes that form the Business Operational View; *refer Chapter 1 Subsection 1.1.3*, a part of the Open-EDI reference model. UMM is divided into three main views to build a so called business collaboration model.

- Business Requirements View (BRV)
- Business Choreography View (BCV)
- Business Information View (BIV)

Figure 5.1 depicts the workflow of UMM modeling process. UMM is a UML profile which is used to model the business collaborations between organizations of a specific business domain. We follow a top-down approach by initially capturing all business requirements in the BRV. In the BCV, the steps required to exchange business documents are modeled here. The flow of business activities involving two participants to complex business transactions or in other words, business collaborations which involves more than two participants, are modelled here. Finally, in the BIV business documents exchanged during the business transactions are modeled using the UN/CEFACT core components through the UML profile for core components. Core components are syntax independent, reusable building blocks that are standardized by the UN/CEFACT for modeling business documents. The core components are defined in the repository Core component Library (CCL) [35]. However, UMM does not mandate which business document standard to use. Core components are just one of many solutions. A brief explanation for each of the modeling views are detailed below.

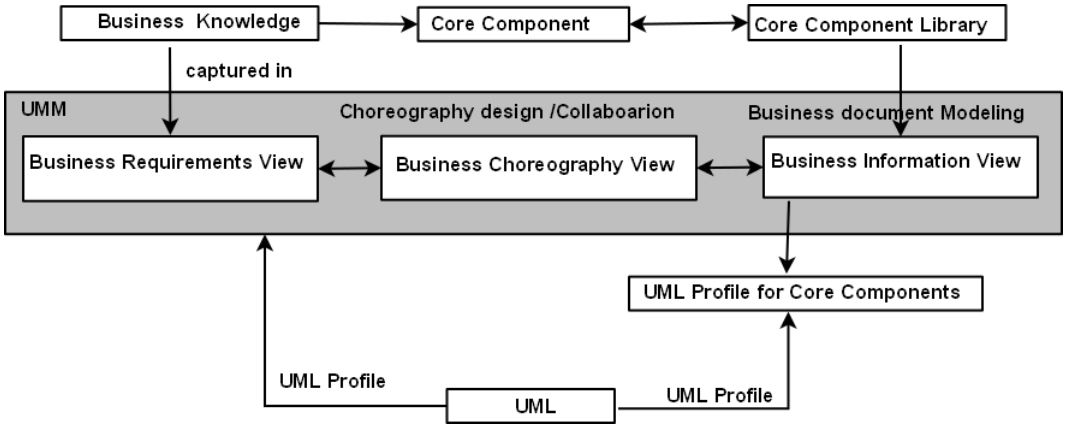

Figure 5.1: Workflow - UMM and UPCC

5.1.1 Business Requirements View

Business Requirements View (BRV) is the starting point where a business analyst tries to gather existing knowledge and issues from business domain experts. The information captured is based on informal textual description. Diverse business activities and problems are analysed and identified here, however, not constructed. The business activities defined, can be internal within an organization or between organizations. Business partners taking part in business processes are also identified here.

The business requirements view is divided into three main sub views:

- *Business Domain View*: Discovers business segments of a business domain and describes the workflow of related business activities. Use case diagrams are used to represent the workflow of related business activities.
- *Business Partner View*: Identifies all participants of take part in the execution of a business activity namely *business partners* and those who just take interest in the workflow of business activities namely *stakeholders*.
- *Business Entity View*: During the execution of business activities, real world objects with independent existence undergo a different stages and are of importance to its participants as well as during the workflow of business activities.

5.1.2 Business Choreography View

Based on the business requirements gathered, the business analyst proceeds towards designing the business collaboration between *business partners*. The *business choreography view* builds upon the *business requirements view* in terms of requirements gathered

for designing inter-organizational business collaborations and synchronisation of *business entities*. Three main sub views within the *business choreography view* are involved in designing the business choreography.

- *Business Transaction View:* Models an information exchange between exactly two participants who play a specific role through a basic building block known as the *business transaction*. A *business transaction* is responsible for coordinating the information systems of both the *business partners*. Coordinating the information systems means that all *business entities* are in the same state in each information system. If a *business entity* changes its state, a *business transaction* is initiated to synchronize with its collaborating *business partner*.

- *Business Collaboration View:* Models complex business interactions through a *business collaboration*. A *business collaboration* involves two or more participants who play different roles in multiple *business transactions* in order to accomplish a shared business goal and terminates upon recognition of an agreed state. A *business collaboration* may collaborate with other *business collaborations* as well.

- *Business Realization View:* A *business collaboration* or *business transaction* are not partner specific. In order to specify that a set of *business partners* collaborate the concept of *business realization* is used. *Business partners* are those already defined in the *business partner view. Business partners* can play different roles in a *business collaboration* thus enabling the concept of reuse within a given scenario.

5.1.3 Business Information View

Business Information View is a container of artifacts that describe the information exchanged in *business collaborations*. UMM suggests using UN/CEFACT's core components for modeling business documents exchanged during *business collaborations*. However, any other method of choice may be taken to describe artifacts in this view. Since core components are UML based, like the *business information view*, syntax independent and standardized by UN/CEFACT, the UML Profile for Core Components (UPCC) is suggested within the *Business Information View*.

In order to define a business document, a modeler takes the generic core components and tailors them to a specific business context and they are called as *Business Information Entities* which form business document artifacts. A *message assembly* is used to aggregate these artifacts to a specific business document. The *message assembly* which is exactly one *information envelope* is used to represent a concrete business message exchanged during a *business transaction*. Business documents created in the *Business Information View* define the exchanged business information on a conceptual level. In

a consecutive step, the conceptual business documents may be transformed to XML schema definitions.

5.2 Worksheets in UMM

Before beginning with the design process of the business process model, necessary details regarding the business environment are noted down. A business analyst does this job by getting into contact with business experts and gathers business information from the business domain of interest. The data captured between a business analyst and business domain experts are first designed as plain text and later modeled and elaborated as UML diagrams in a UMM tool of choice. However, if UMM models were described as UML syntax in one modeling tool and the plain text description in another, it would be complicated to interact and can lead to a lot of inconsistencies. Hence, UN/CEFACT has proposed the idea of so called worksheets for UMM that summarizes all the information represented as UML diagrams in a textual format for different artifacts created during the modeling process thus persisting business domain knowledge. The worksheets do the job of collecting business information and requirements in a plain text language. These worksheets assist a business analyst while capturing business agreements, requirements and all necessary details required for modeling the business process model.

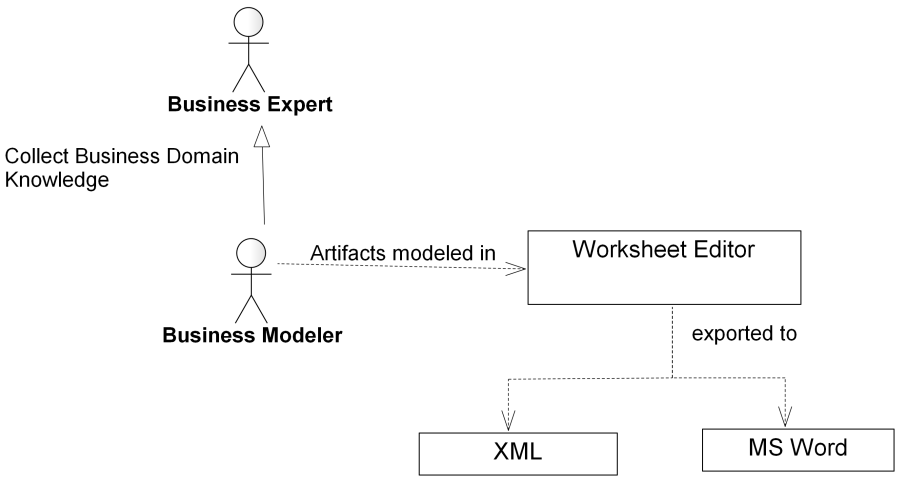

Figure 5.2: Workflow - WorksheetEditor

In UMM, a predefined set of stereotypes are represented as worksheets for modeling purposes. In UMM 2.0, nine pre-defined worksheets for different stereotypes are

elaborated in *Appendix - Worksheets*. The modeler is not confined to these stereotypes represented as worksheets but has the freedom to choice to represent the stereotypes of his choice as worksheets.

Tagged values and constraints represent the content of worksheets. Since the business requirements vary from one business domain to another, the modeler has the freedom to edit the layout by confining or extending the content of the worksheets. This way, the modeler can customise the tagged values and constraints in UMM captured in the worksheets based on the business domain. Not every stereotype is created as a worksheet. The content of one stereotype might be captured in its parent stereotype that already contains a worksheet or all the contents are captured and detailed in the worksheet of the child stereotype. Hence, while modeling, the modeler needs to analyse which stereotype needs to be reflected as a worksheet. Each stereotype has its own distinct representation as a worksheet based on its tagged values and constraints.

Consider taking the worksheet for stereotype *business process* shown in *Appendix - Worksheets* into consideration. In the example `OrderFromQuote`, the *business process*, `Place Purchase Order` in Table 5.4 (refer Section: Business Requirements View, Subsection: Business Domain View) contains blocks `General, Details` and `Relationships` representing the tagged values in UMM. Additionally, `Start/End Characteristics` block represents the constraints in UMM. A *business process* stereotype is defined from its parent stereotype *business process use case* for which no worksheet is represented.

The worksheets act as an bridge between business experts and business modelers. Additionally, it is possible to document the information collected in the worksheet by exporting them as word or XML file as shown in Figure 5.2. The default structure of the worksheets is based on an XML schema, through which xml files is defined. The end report, provides business experts taking part or interest in business processes, the exact information captured in worksheets.

5.3 Business Collaboration Model

5.3.0.1 Overview and purpose

A model is a simplified version or representation of some aspect, defined in the context of a domain of interest. In business settings, modeling workflows and processes aren't just internal - they're also inherently cross-organizational. Even a simple business process such as ordering a product, involves more than one autonomous party. Modeling a collaborative business process, takes a mediating role that binds cross-organizational business processes. In UMM, collaborative business process model is termed as a *business collaboration model*.

5.3.0.2 Step-by-Step modeling guide

A *business collaboration model* which is UMM compliant, is a package that extends the *business library* from the UMM base module and is stereotyped as *BusinessCollaborationModel (bCollModel)*. A *business collaboration model* can contain three different views:

1. Business Requirements View (BRV)
2. Business Choreography View (BCV)
3. Business Information View (BIV)

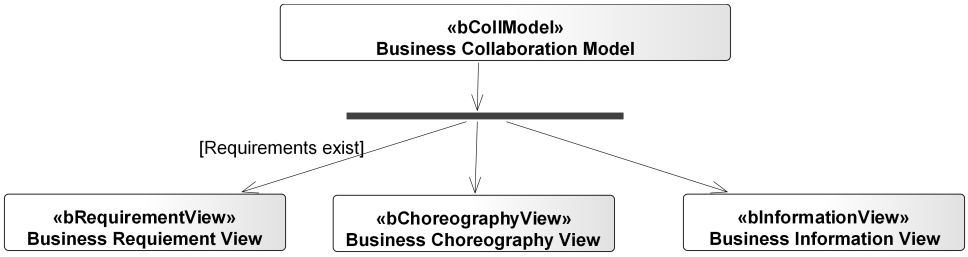

Figure 5.3: Workflow - Business Collaboration Model

The Business Requirements View, a package stereotyped as the *BusinessRequirementsView (bRequirementsV)* is used to gather business knowledge from stakeholders and business experts if business requirements exist. A business collaboration model may have any number of *bRequirementsV* or no *bRequirementsV* at all. The Business Choreography View, is a package stereotyped as the *BusinessChoreographyView (bChoreographyV)* is used to design a sequence of events required for business information to be exchanged between business partners. The Business Information View, a package stereotyped as the *BusinessInformationView (bInformationV)* is used to model business documents exchanged during the business collaboration in the *bChoreographyV*. A business collaboration model must have at least one *bChoreographyV* and a *bInformationV* defined as shown in Figure 5.3 .

Figure 5.4, shows a tree view representation of the business collaboration model `OrderFromQuote`. This model contains three packages stereotyped as *bRequirementsV*, *bChoreographyV* and *bInformationV*.

5.3.0.3 Tagged values of stereotypes within the *bCollModel*

Stereotypes *bCollModel*, *bRequirementsV*, *bChoreographyV* and *bInformationV* are based on its parent stereotype *bLibrary* (from the UMM base module) and hence inherit all the following mandatory tagged values:

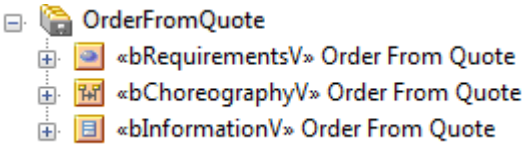

Figure 5.4: Business Collaboration Model - OrderFromQuote

- *uniqueidentifier*: A unique identifier represents the *business library* package in the registry. A UUID (Universally Unique Identifier) is used within the URI structure scheme to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library*.
- *versionidentifier*: The version identifier represents the current version of the *business library* in the registry. The version information should not be assigned by the modeler rather must be managed in the registry. An unregistered library must not have any version assigned.
- *businessTerm*: A synonym frequently used for the stereotype in business.
- *copyright*: Copyright of the *business library* package.
- *owner*: The owner of the *business library* which could be an organization, institution or an individual.
- *reference*: Contains location information to additional resources where further information about the *business library* can be found.
- *status*: It represents the current lifecycle status of the registered *business library* package. The status is set by the registry. The possible status representing the stereotype are *proposed, approved, mandatory, validated* and *implemented*. A registry unregistered must not have any status information assigned.

The *bCollModel* defines an additional tagged value *justification* which explains from the business perspective the reason why a business case is considered for possible business collaborations.

5.4 Business Requirements View

5.4.0.1 Overview and purpose

At the beginning of any business choreography modeling process, it is necessary to understand the requirements of business experts and stakeholders, in order to overcome the mismatch between what has been designed and what is actually needed. In a *business*

requirements view, a modeler captures requirements in the language which is familiar to the modeler as well as from whom the requirements are gathered. The result of the *business requirements view* should not be the construction of new business logic rather discovering the existing ones.

In this view all existing business events (e.g. ordering a quote, placing an order) containing a series of structured business activities which could be internal within an organization or between organizations are discovered. Additionally, all the participants and stakeholders from the business domain are identified in this view. Business events result in changing the state of real-world objects and/or a product having business significance. These objects and their changes in state are also modeled in this view. A simple example would be an "Order" that undergoes different stages when it is submitted and processed.

5.4.0.2 Step-Step modeling guide

The *BusinessRequirementsView* contains three views that serve distinctive purposes:

1. Business Domain View (BDV)
2. Business Partner View (BPV)
3. Business Entity View (BEV)

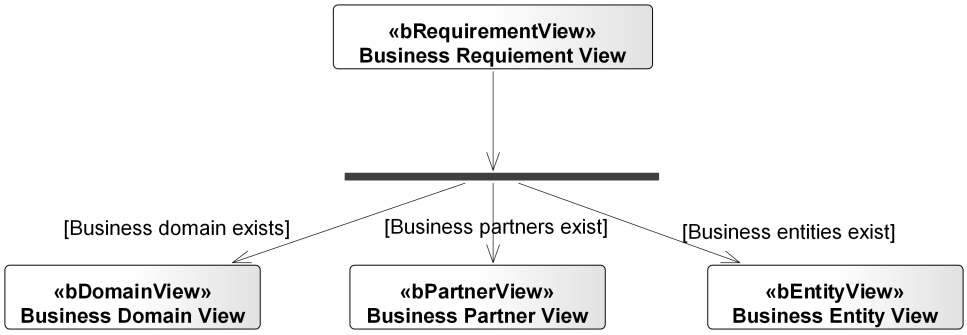

Figure 5.5: Workflow - Business Requirement View

Figure 5.5 shows an activity diagram based on how the *business requirements view* is modeled. The Business Domain View is a package stereotyped as *BusinessDomainView (bDomainV)*. It is used to discover and design the flow of existing business logic of a business domain of relevance. The Business Partner View is a package stereotyped as *BusinessPartnerView (bPartnerV)*. It lists the business partners and stakeholders taking part or interest in business activities of a business domain under consideration.

The *BusinessRequirementsView* can contain maximum only one *BusinessDomainView* and *BusinessPartnerView*. The Business Entity View is a package stereotyped as *BusinessEntityView (bEntityV)*. It contains a list of real-world objects and the changes in state they undergo due to the execution of business activities. Any number of *BusinessEntityViews* can be modeled within a *BusinessRequirementsView*.

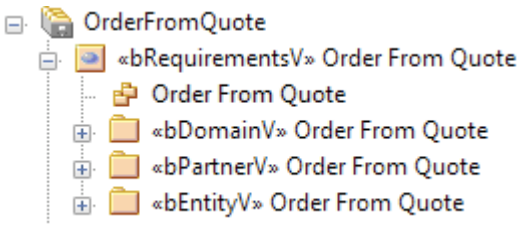

Figure 5.6: Business Requirements View - OrderFromQuote

Figure 5.6 shows a tree view representation of the *bRequirementsV*. Three stereotypes namely the *bDomainV, bPartnerV and bEntityV* are created beneath the *bRequirementsV*.

5.4.0.3 Tagged Values of stereotypes within the *bRequirementsView*

The stereotypes *bDomainV*, *bPartnerV* and *bEntityV* are based on its parent stereotype *bRequirementsV* whose parent stereotype is in turn the *bLibrary* (from the UMM base module). Therefore, all the following mandatory tagged values are inherited:

- *uniqueidentifier*: A unique identifier represents the *business library* package in the registry. A UUID (Universally Unique Identifier) is used within the URI structure scheme to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library.*
- *versionidentifier*: The version identifier represents the current version of the *business library* in the registry. The version information should not be assigned by the modeler rather must be managed in the registry. An unregistered library must not have any version assigned.
- *businessTerm*: A synonym frequently used for the stereotype in business.
- *copyright*: Copyright of the *business library* package.
- *owner*: The owner of the *business library* which could be an organization, institution or an individual.
- *reference*: Contains location information to additional resources where further information about the *business library* could be found.

- *status*: It represents the current lifecycle status of the registered *business library* package. The status is set by the registry. The possible status representing the *business requirements view* are *proposed, approved, mandatory, validated* and *implemented*. An registry unregistered must not have any status information assigned.

5.4.1 Business Domain View

5.4.1.1 Overview and purpose

The *business domain view* relies upon the *business requirements view*. The knowledge gathered between a modeler and the business experts includes various business characteristics, procedures, requirements and features of the business domain. The task of the business domain is to categorize and disclose business activities using a predefined classification scheme. Organizing related business activities enables the modeler to achieve a clear and transparent view of the a complex business domain. Business activities that involve one or more participants are described in this view from a global point of view.

Table A.1 in the Appendix shows the details gathered for the business domain view worksheet. Table 5.1 displays the content of the worksheet for the *business domain view* `OrderFromQuote`. The information gathered, gives a detailed information about the business domain, registered information of the *business library*, a list of *business areas* or *business category* of this business domain.

Business Domain View	
General	
Name	OrderFrom Quote
Description	This business domain view outlines and categorizes all business process events discovered for ordering a quote between a purchasing and a selling organization. When a new customer registers itself at the selling organization then an account is created for the customer, requesting a credit check. The customer of the purchasing organization receives a customer id, request the price of quotes, in turn receives the quote products and finally places a purchase order. During these series of actions, the bank is used to provide the credit results Ultimately, the product is sold by the selling organization.
Business Library Information	
URI	
BusinessTerm	Quote to Order, Purchase Order, Request for Quote (RFQ), Order, Sales Order, Price Request
Version	1.0
Status	Approved
Owner	UN/CEFACT
Copyright	UN/CEFACT
Reference(s)	- Purchasing Organization - Selling Organization
Business Area(s) (add columns as needed)	
Business Area	- Financial Areas - Procurement/Sales

Table 5.1: Business Domain View Worksheet - OrderFromQuote

5.4.1.2 Step-by-Step modeling guide

In order to classify a set of related business activities, UMM uses the concept of *business areas* and *process areas*. In case a modeler finds it difficult to categorize a *business process*, the concept of *business categories* are used.

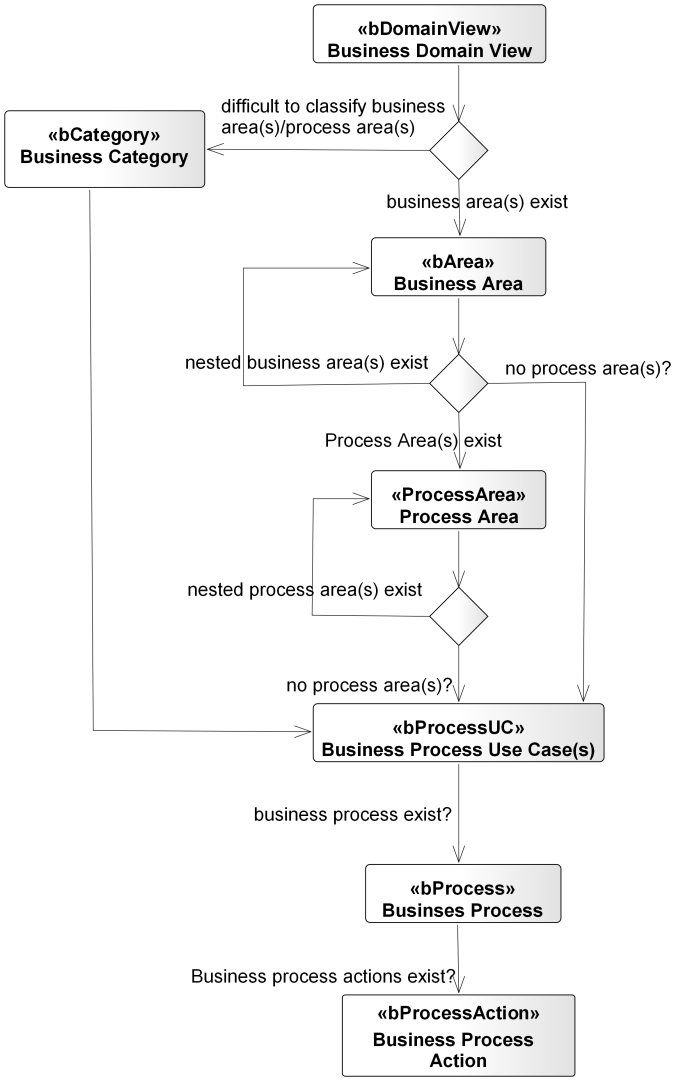

Figure 5.7: Workflow - Business Domain View

Step 1. ***Discover and create business areas within a business domain***

Overview and purpose

A *business area* is an organizational unit corresponding to a specific business segment within an enterprise. In order to elaborate the layout structure, all relevant *business areas* of a business domain are identified. UMM does not mandate a precise classification of *business areas*. However UN/CEFACT Com-

mon Business Process Catalogue [47] suggests a list of eight non recursive categories: *Procurement/Sales, Design, Manufacture, Logistics, Recruitment, Training, Financial Services, Regulation and Health Care*. This list of *business areas* is considered as non-exhaustive.

Model a business area

A *business area* stereotyped as *BusinessArea (bArea)* is a package and is created within a *business domain view*. At least one *business area* should be defined within a *business domain view*. Within a *business area*, nested *business areas* can be discovered and created.

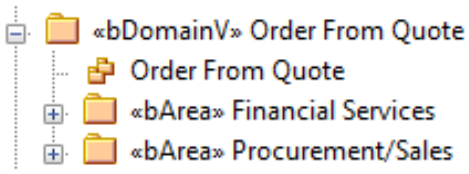

Figure 5.8: Business Domain View - Order From Quote

Figure 5.8, shows two *business areas* - `Procurement/Sales` and `Financial Services` identified beneath the *business domain view* `Order From Quote`. Table 5.2 shows the worksheet for the *business area* `Procurement/Sales`. The information gathered here includes a description of the *business area* `Procurement/Sales`, details regarding the *business area*, registered information in the *business library,* as well as a list of its child *business area* and *processes areas* of the *business area* `Procurement/Sales`.

Business Area	
General	
Name	Procurement/Sales
Description	The *business area* - Procurement/Sales discovers all business processes by grouping them to its respective *process areas* - Identification and Negotiation.
Details	
Objective	This business sector provides the requested quote to its customer and delivers its product in a timely efficient manner.
Scope	Public
Business Opportunity	Sells the product to its customer based on the requested quote of products
Included in	Order From Quote
Business Library Information	
URI	
BusinessTerm	Quote to Order, Purchase Order, Request for Quote (RFQ), Order, Sales Order, Price Request
Version	1.0
Status	Approved
Owner	UN/CEFACT
Copyright	UN/CEFACT
Reference(s)	
Business Area(s)	
Business Area No 1..n	None
Process Area(s)	
Process Area No 1..n	- Identification - Negotiation

Table 5.2: Business Area Worksheet - Procurement/Sales

Step 2. ***Discover and create process areas within a business area***

Overview and purpose

A *process area* is a cluster of related business activities executed within a *business area*. Any number of *process areas* can be identified within a *business area*. However, it is recommended to use a list of *process areas* identified by the UN/CEFACT Common Business Process Catalogue [47]: *Planning, Identification, Negotiation, Actualization, and Post Actualization*. This list of flat (non-recursive) categories represents the five phases of the business collaboration as defined in the ISO Open-EDI model [6].

Model a process area

A *process area* stereotyped as *ProcessArea* is a package and is created within a *business area* of the *business domain view*. A lowest level of a *business area* can contain any number of *process areas* or no *process areas* at all. Nested *process areas* within a *process area* can also be discovered. In case, no *process areas* are found within the lowest hierarchy of a *business area* or no nested *process areas* are discovered, then *business process use cases* can be described that exist within the scope of the *business area* or *process area*; refer Step 4. It is possible to have *process areas* as well as *business process use cases* on the same level hierarchy of a *business area*.

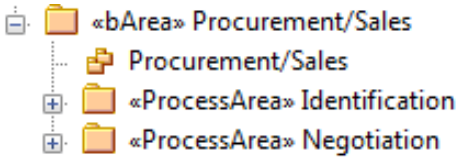

Figure 5.9: Business Area - Procurement Sales

In our example, within the *business area* `Procurement/Sales` two *process areas* `Identification` and `Negotiation` are identified as shown in Figure 5.9. Within the *process area* `Negotiation`, one nested business *process area* `Inter-organizational Process (to-be)` is discovered as shown in Figure 5.10

Table A.3 in the Appendix, shows the details required to be captured for the *process area* worksheet template and Table 5.3 shows the data gathered for the *process area* `Negotiation` worksheet. The information gathered here includes a description of the *process area* `Negotiation`, details of the *process area*, information registered in the *business library* and as well as a list of its child *process areas*.

Process Area	
General	
Name	Negotiation
Description	This *process area* describes *business processes* that include requesting the price quote for required products, processing the request for the quote followed by placing the purchase order for the desired product, processing the order and based on the decision confirm the processed order.
Details	
Objective	Complete a deal between the purchasing organization as well as the selling organization.
Scope	Public
Business Opportunity	Sell the product to its customer based on the requested quote of products
Included in	Procurement/Sales
Business Library Information	
URI	
BusinessTerm	Quote to Order, Purchase Order, Request for Quote (RFQ), Order, Sales Order, Price Request
Version	1.0
Status	Approved
Owner	UN/CEFACT
Copyright	UN/CEFACT
Reference(s)	
Process Area(s) (if present or add additional process areas if needed)	
Process Area No 1..n	- Inter-organizational Process (to-be)

Table 5.3: Process Area Worksheet - Negotiation

Step 3. ***Discover and create business categories within a business domain (alternative for step 1 and step 2)***

Overview and purpose

If a modeler finds it difficult to classify *business areas* and *process areas* within a business domain, UMM provides the opportunity of defining an abstract concept, namely *business category*. Related *business processes* identified are defined in its respective business categories thereby enabling the classification of *business processes*.

Model a business category

A *business category*, stereotyped as *BusinessCategory (bCategory)* is created

beneath the *business domain view*. A *business category* is an alternative classification schema to *business areas* and *process areas* that classifies *business processes* within a *business domain view*. It can also be used to group other *business categories*. As an alternative, *business processes* that belong to the respective *business category* can be grouped as well.

The *business area* worksheet can be used as a template to gather the requirements of a *business category*. The information gathered includes a description of the *business category*, registered information in the *business library* and a list of *business categories* grouped under the *business category* taken into consideration.

Step 4. ***Discover and create business processes use cases within a business area or a process area.***

Overview and purpose
A *business process use case* summarizes the behaviour of *business processes* along with its dependencies yielding a significant value to its related business parties such as *stakeholders* and *business partners*. Within a *business area* or *process area*, business processes are discovered and classified as *business process use cases*. A *business process use case* is realized by *business processes*.

Describe a business process use case
A *business process use case* stereotyped as *BusinessProcessUseCase (bProcessUC)* is created within a *business area* or *process area* or a *business category* of a business domain. A *business process use case* is described with the help of a sequence diagram or a use case diagram. More than one use case diagram can be used to facilitate the modeling of a *business process use case*. A *business area*, *process area* or a *business category* can contain any number of *business process use cases* or no *business process use cases* at all.

A *business process use case* is described with the help of *business partners* and *stakeholders* who take part or interest in *business processes*. These participants are discovered and defined in the *business partner view*; *refer Section 5.4.2)*. In a *business process use case* diagram, *business partners* are associated to the *bProcessUC* via UML association stereotyped as *participates*. Identified *stakeholders*, on the other hand are associated to the *bProcessUC* via UML dependency, stereotyped as *isOfInterestTo*. At least one *business partner* must exist within a *business process use case* diagram and any number of *stakeholders* or no *stakeholders* at all. A *stakeholder* can take interest in multiple *business process use cases*. On the other hand, a *business partner* can also take part in multiple *business process use cases*.

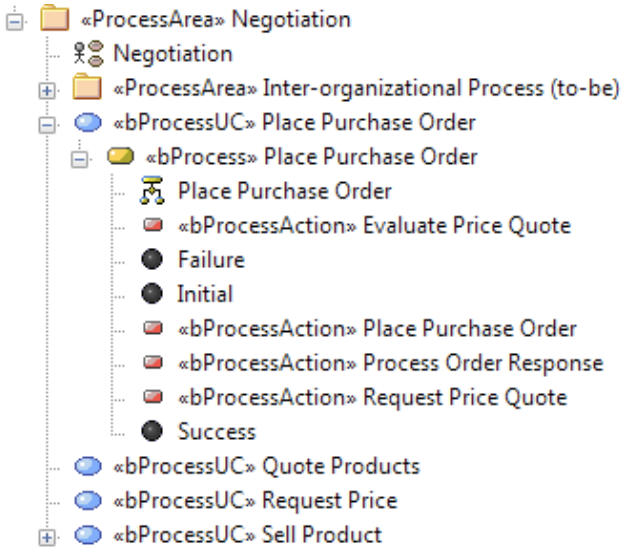

Figure 5.10: Process Area - Negotiation

In our example as shown in Figure 5.10, within the *process area* `Negotiation`, four *business process use cases* are identified: `Place Purchase Order`, `Quote Products`, `Request Price` and `Sell Product`.

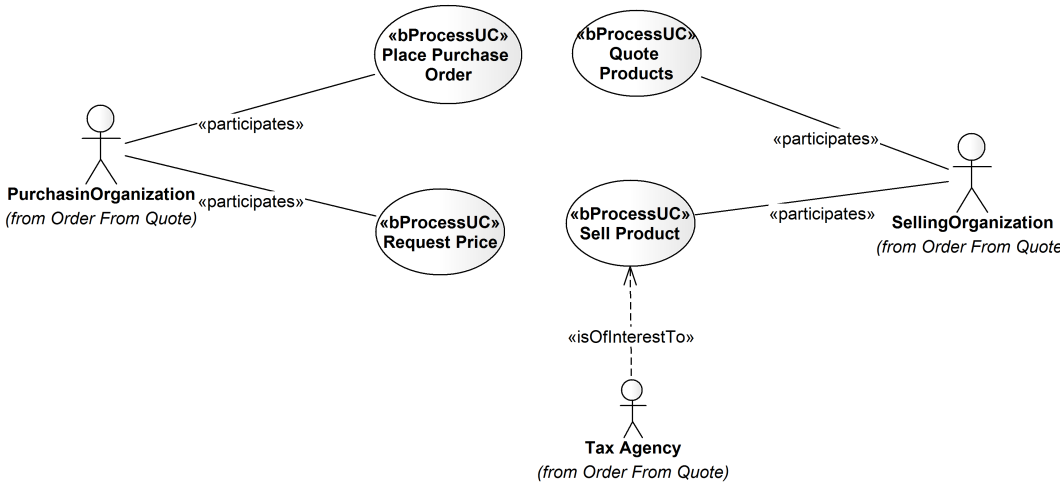

Figure 5.11: Process Area Use Case Diagram - Negotiation

In Figure 5.11, the *business partner*, `Purchasing Organization` participates in *business process use cases* `Place Purchase Order` and `Quote`

Products. The Selling Organization participates in the *business process use cases* Request Price, Sell Product. The *stakeholder*, Tax Agency is interested in the result of in the *business process use case* Sell Product and is connected via *isofInterestTo* association.

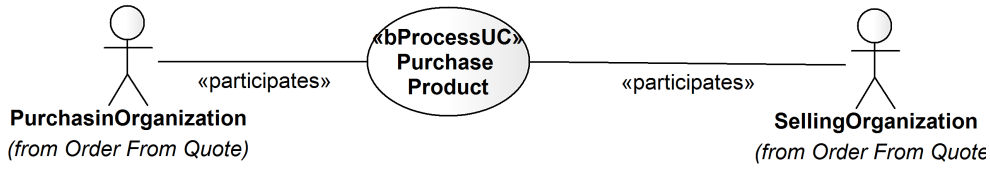

Figure 5.12: Business Process Use Case Diagram- Inter-organizational Process (to-be)

In Figure 5.12, taking the nested *process area* Inter-organization Process(to-be) into consideration, within the *process area* Negotiation, a *business process use case* is defined. *Business partners*, Purchasing Organization and the Selling Organization participate in the *business process use case* Purchase Product and are associated via *participates* association.

Step 5. ***Create a business process within a business process use case.***

Overview and Purpose
A *business process* represents a dynamic behavior of a *business process use case*. A *business process* is a set of business activities aiming to accomplish a certain goal to its related parties. *Business processes* that involve only one business partner are known as internal *business processes*. On a contrary, *business processes* involving two or more *business partners* are called inter-organizational *business processes*. A *business process* does not have a specific structure defined in order to describe its flow of activities, since different ways exist in designing a *business process*. Hence, it is most appropriate for the business analyst to study the flow of activities and identify *business processes* with the help of business experts.

Create a business process
A *business process* stereotyped as *businessProcess (bProcess)* is created beneath the *business process use case* and represented in the form of a UML activity diagram. A *business process use case* can contain multiple *business processes*, each representing different flow of business activities.

Step 6. ***Create business processes actions within a business process.***

Overview and purpose
The flow of business activities in a *business process* are constructed using the concept of *business process actions*. Every *business process action* represents a step of execution in the *business process*.

Describe the flow of a business process
A *business process action* stereotyped as *BusinessProcessAction (bProcessAction)* is created beneath the *business process*. A *business process action* may be composed of nested *business processes*. Thereby, a *business process action* can also be refined within another *business process*.

In a *business process* execution, there are real world things that have business values shared among their participants. These objects are called *business entities*. A *business entity* undergoes different stages during business process execution. The behaviour of *business entities* are modeled in the *business entity view*; *refer Section 5.4.3*. Execution of a *business process action* results in a change of a *business entity state*. A change of a process's state can happen due to a result of completing an activity or also in other non-anticipated ways. For example, a customer may change, or cancel the order submitted. The *business entity state* is a result of the preceding workflow of a *business process* and determines in turn the further workflow. Hence, a *business entity state* is the output of a *business process action* or another *business entity state* and at the same time an input to the following *business process action* or another *business entity state*. Since business entities highly impact the business execution process, modeling of a *business process* and a *business entity view* are highly dependent upon each other.

In the *Appendix-Worksheets*, Table A.4 shows the *business process* worksheet template and Table 5.4 shows the information entered into the worksheet for the *business process* `Place Purchase Order`.

The informaton includes a description of the *business process*, details regarding the *business process*, *business areas* and *process areas* to which this *business process* are classified as well as the participating parties and *stakeholders* are stated here. Relationships to other related *business processes* and *business entities* affected during the execution of these *business processes* are stated down. Additionally, pre- and post-conditions and start/end characteristics of the *business process* are stored in the worksheets.

Business Process	
General	
Name	Place Purchase Order
Description	All business events are executed to have the order placed .
Details	
Classified to Business Areas and Process Areas	- Procurement/Sales - Negotiation
Participants and their interests	Purchasing Organization - interested in evaluating and purchasing a price quote.
Stakeholders and their interests	None
Reference(s)	
Start/End Characteristics	
Pre-condition	- Customer is registered in the system - Credit Check is provided - Availability of goods
Post-condition	Order is confirmed or rejected
Begins When	The price of desired quotes is requested and a list of ordered quotes are provided.
Ends When	The order is processed by the selling organization and the purchasing organization receives a confirmation if the order is accepted or rejected
Actions	The order is processed by the selling organization
Exceptions	Non-availability of requested products.
Relationships	
Included Business Processes	None
Affected Business Entities	Order.

Table 5.4: Business Process Worksheet - Place Purchase Order

Related parties who take part in a *business process* are identified as *business partners* and modeled in the *business partner view*; *refer Section 5.4.2*. One or more *business partners* can take part in the execution of the *business process*. UMM distinguishes two ways of modeling *business processes* based on the number of *business partners* involved during execution.

a) ***Business process modeling for a single business partner***
Internal *business processes* that involve just a single *business partner* are represented as a flow of *business process actions* within an activity diagram or within a single UML partition of an activity diagram. *Business entity states* manipulated within the workflow of a single *business partner*

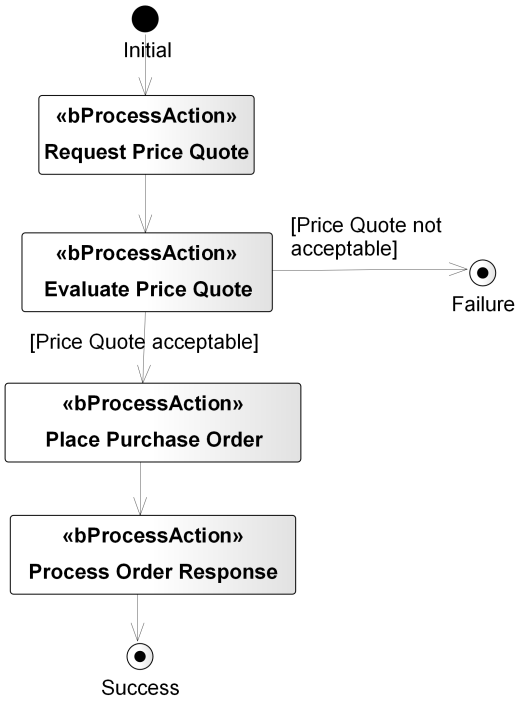

Figure 5.13: Business Process Activity Diagram - Place Purchase Order

are called internal business entity states which are object nodes stereotyped as *InternalBusinessEntityState (bEInternalState)*. *Business process actions* influenced during the transition of internal *business entity states* exist within the same UML partition. A *business process* can contain *business entities* undergoing any number of *internal business entity states*. *Internal business entity states* can also exist in an business process execution of a *business partner* in an inter-organizational *business process*.

An example for an internal business process execution is shown in Figure 5.13. The *business process* `Place Purchase Order` for the *business partner* `Purchasing Organization` is described with the help of *business process actions*: `Request Price Quote`, `Evaluate Price Quote`, `Place Purchase Order` and `Process Order Response`.

An ideal example of internal *business entity states* is demonstrated in Figure 5.14.Within the internal business process execution of the *business partner* `Selling Organization`, the execution of *business process action* `Process Request of Quote` results in a change of state

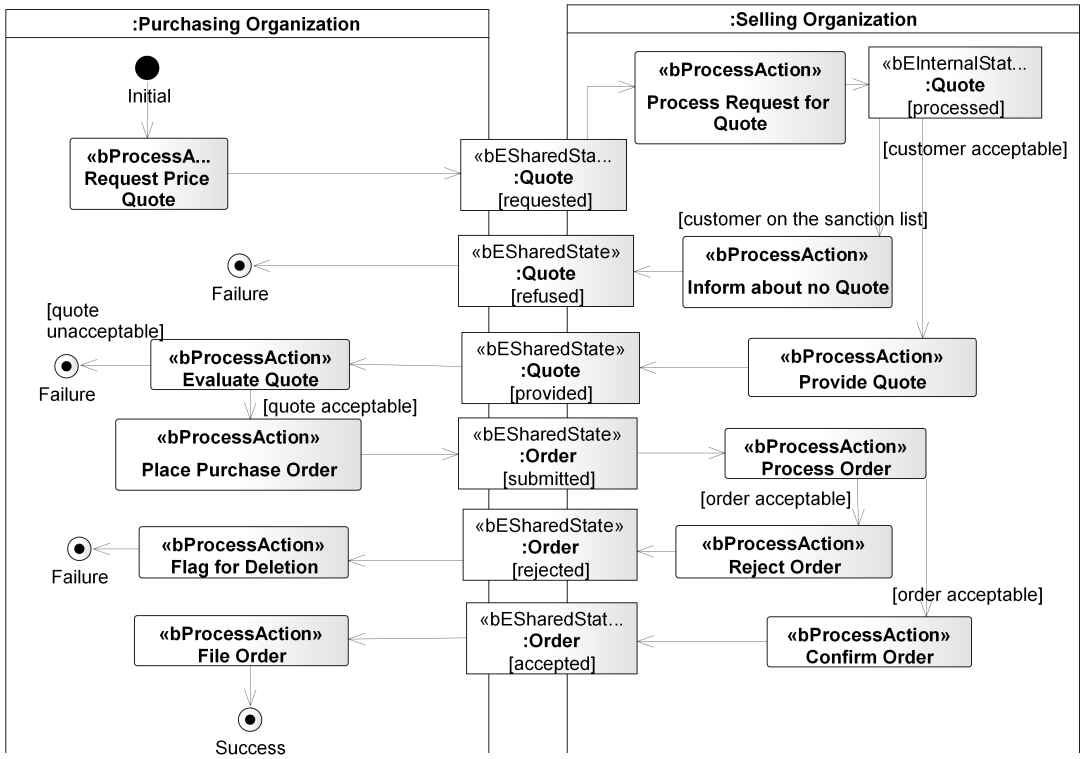

Figure 5.14: Business Process Activity Diagram - Inter-organizational Process (to-be)

from `requested` to `processed` of business entity `Quote`. This change of *business entity state*, renders the execution of two possible *business actions* - `Provide Quote` and `Inform about no Quote`.

b) ***Business process modeling between two or more business partners***

For *business processes* involving two or more *business partners*, UMM uses the concept of UML partitions. Each business activity partition represents a *business partner*. Each business activity partition describes the flow of business activities performed by its respective *business partner*.

If two consecutive *business process* actions are executed between two *business partners*, a state of change occurs before the execution of the following process. Hence, a transition is created from a *business process action* of the first *business partner* to a *business entity state* signalling the output. At the same time, a transition from the same *business entity state* to the *business process action* of the second *business partner* is created signalling an input. *Business entity states* manipulated between the workflow containing more than one *business partner* are known as *shared business*

entity states which are object nodes stereotyped as *SharedBusinessEntityState (bESharedState)*. *Business process actions* influenced during the transition of *shared business entity state* exist in two different UML activity partitions. An inter-organizational *business process* can have *business entities* that undergo any number of *shared business entity states* during the execution.

In our example as shown in Figure 5.14, the `Inter-organizational process` activity diagram shows the activities performed between two *business partners* `Purchasing Organization` and `Selling Organization` represented as UML partitions. Each of these partitions are assigned as classifiers, referring to its *business partners*. Within the *business partner* `Purchasing Organization`, internal *business process actions* such as `Request Price Quote`, `Evaluate Quote`, `Place Purchase Order`, `Flag For Deletion`, `File For Order` are executed. Similarly, the *business partner* `Selling Organization` executes *business process actions* such as `Process Request For Quote`, `Inform about no Quote`, `Provide Quote`, `Process Order`, `Reject Order`, `Confirm Order` to complete the business process execution.

Taking the *business partner* `Purchasing Organization` into consideration, after the execution of *business process action* `Request Price Quote`, a change of state of the *business entity* `Quote` is noticed from its initial state to `requested`; *see Figure 5.19, refer Section 5.4.3*. This change of state initiates the execution of the *business process action* `Process Request of Quote` of the *business partner* `Selling Organization`. Since, the *business entity state* `requested` manipulates the state of execution in both *business partners*, it is defined as a *shared business entity state*.

5.4.1.3 Tagged Values of stereotypes within the *bDomainView*

The stereotype *bCategory* contains the following tagged values. Stereotypes *bArea* and *process area* inherit all attributes from *bCategory*.

- *uniqueidentifier*: A unique identifier represents the *business library* package in the registry. A UUID (Universally Unique Identifier) is used within the URI structure scheme to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library.*

- *versionidentifier*: Version identifier is the current version of the *business library* in the registry. The version information should not be assigned by the modeler

rather must be managed in the registry. An unregistered library must not have any version assigned.

- *businessTerm*: A synonym frequently used for the *business process* in business.
- *copyright*: Copyright of the *business library* package.
- *owner*: The owner of the *business library* which could be an organization, institution or an individual.
- *reference*: Contains location information to additional resources where further information about the *business library* could be found.
- *status*: The status represents the current lifecycle status of the stereotype registered in the *business library* package. The status is set by the registry. The possible status representing the stereotype are *proposed, approved, mandatory, validated* and *implemented*. An registry unregistered must not have any status information assigned.
- *objective*: The goal intended to be attained by the *business area*, *process area* or the *business category* under consideration.
- *scope*: Defines the limit of access of the *business area*, *process area* or the *business category* within the *business collaboration model* and is defined as *public, private or protected*.

The stereotype *participates* is an association containing a tagged value *interest* that states the purpose of interest of a *business partner* in a *business process use case*.

The stereotype *isOfInterestOf* is a dependency with *interest* as a tagged value that states the purpose of interest of a stakeholder in a *business process use case*.

The stereotype *bProcessUC* contains the following attributes:

- *definition*: describes the purpose of a *business process use case*.
- *preCondition*: A set of conditions that need to be fulfilled before the execution of a *business process*.
- *postCondition*: A set of conditions that need to be fulfilled after the execution of a *business process*.
- *beginsWhen*: Specifies certain business events that trigger the initiation of the *business process*. It could also be a case where a certain state needs to be specified in order to begin a *business process*.
- *endWhen*: Specifies a list of business events or conditions that lead to the termination of a *business process*.
- *actions*: Describes the activities that are performed during the execution of a *business process*.

- *exceptions*: Lists all exception conditions that causes the *business process* to terminate before its normal completion.

5.4.1.4 Summary of artifacts created within the *bDomainView*

As far as of now, a modeler is presented a step-by-step procedure required towards gathering a basic understanding of the business processes in a business domain. *Business processes* discovered are classified in the *business domain view*. Results of the artifacts created within the *business domain view* include:

- ***business category***: An abstract concept that identifies and classifies *business processes* of a business domain. It can be used to group other *business categories* or *business processes* from another *business category* as well.
- ***business areas***: Small units within an enterprise. They can be recursively composed of *businessAreas* or *processAreas*. UMM does not mandate a specific classification schema however it does suggest the use of eight flat categories from the UN/CEFACT common business process catalogue [47].
- ***process areas***: Groups of related business activities belonging to its respective *business area*. A *process area* can be recursively composed of *processAreas* or *business process* use cases. UMM does not mandate a specific classification schema however it does suggest the use of five flat categories from the UN/CEFACT common business process catalogue [47] to the five successive phases of business collaboration.
- ***business process use case diagrams***: Represent business events executed by *business partners*. It can be executed by a single business partner or in cases of crossing organizational boundaries between multiple *business partners*. In a collaborative business environment, a *business process use case* should create a value to all its participants.
- ***business processes***: Activities that realize *business process use cases* by describing the execution of the *business process*.
- ***participates***: An association between a *business partner* and a *business process use case*. This association indicates that an input or output is provided by the *business partner* to its associated *business process use case*.
- ***isOfInterestTo***: A dependency from the *business process use case* to a stakeholder. The interest of the stakeholder in the *business process* is defined in this artifact.
- ***business process action***: A step in the execution of a *business process*. A *business process* can also be refined by another *business process*.

- ***internal business entity state***: An object that represents a *business entity state* that is internal to a *business process* of a *business partner*.
- ***shared business entity state***: An object that represents a *business entity state* between two partners in a *business process*.

5.4.2 Business Partner View

5.4.2.1 Overview and purpose

In any *business process* execution, roles played by participants who interact in a *business process* are discovered, those who provide an input and those who are interested in its output. The represented role could be an individual, group or an organization that participate or take interest in a *business process*. The *business partner view* contains a list of users who represent specific roles. In UMM, users take the role of *stakeholders* or *business partners*.

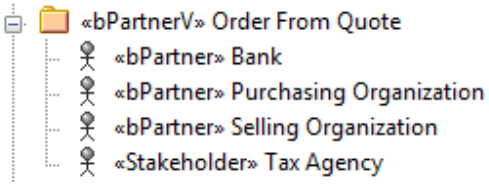

Figure 5.15: Business Partner View - Order From Quote

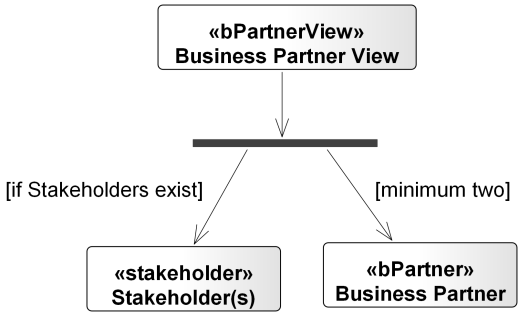

Figure 5.16: Workflow - Business Partner View

5.4.2.2 Step-by-step modeling guide

Step 1. ***Discover and create stakeholders within the business partner view***

Overview of purpose of stakeholders
A *stakeholder* is a person or a representative, interested in the outcome of a *business process*. A *stakeholder* does not necessarily need to take part in the execution of any *business process*.

Model stakeholders
A *stakeholder* is an actor stereotyped as *stakeholder*, created beneath the *business partner view*. Any number of *stakeholders* can be defined within the *business partner view* and can take interest in a *business process*. Once the *stakeholders* are identified they are used in the modeling process of the *business process use case* diagram. In Figure 5.15, the stakeholder, `Tax Agency` defined in the *business partner view* takes interest in a *business process* `Sell Product` as show in Figure 5.11.

Step 2. ***Discover and create business partners within a business partner view.***

Overview of purpose of business partners
A *business partner* could be an organization type, an organization unit type, an institution or a person type who takes part in the execution of *business processes*. *Business partners* are identified when *business processes* are discovered. Since *business partners* take part in a *business process* execution they have a certain amount of interest in the *business process*. Therefore, a business partner is also a special kind of a *stakeholder*.

Model business partners
A *business partner* is an actor stereotyped as *BusinessPartner (bPartner)* created beneath the *business partner view*. At least two *business partners* need to be defined within the *business partner view* and at least one *business partner* participates in the execution of a *business process*. In Figure 5.15, *business partners* - `Purchasing Organization` and `Selling Organization` take part in a *business process* as shown in the *business process use case* diagram - `Negotiation` in Figure 5.11.

5.4.2.3 Tagged Values of stereotypes within the *bPartnerView*

The stereotype *stakeholder* contains a tagged value *interest* which states the purpose of the interest in a respective *business process*.

The stereotype *bPartner* generalizes *stakeholder*, thereby inheriting the tagged value *interest*.

5.4.2.4 Summary of artifacts created within the *bPartnerView*

All *business partners* of a business domain are discovered in the *business partner view*. Results of the artifacts created within the *business partner view* include actors such as

- ***business partners***: Actors which are individuals, organization types or organizational unit types participating in *business processes*. *Business partners* provide an input and receive an output from the execution of *business processes*. Since *business partners* take part in the execution of *business processes* they also have a vested amount of interest in a *business process*. Hence, a *business partner* is also a special kind of a *stakeholder*.
- ***stakeholders***: Actors who are individuals or representatives of an organization having a vested interest in a outcome of a *business process* or a *business category*. *Stakeholders*, however do not necessarily participate in the execution of *business processes*.

5.4.3 Business Entity View

5.4.3.1 Overview and purpose

When discussing *business processes*, business relevant real objects are created and evolved during the execution. These objects and the information they carry reflect the quality of the overall business process. Hence, they are refered as *business entities*. An example would be a financial deal in a bank going through states such as Draft, Offered, Signed, Active etc. In the *business entity view*, a *business entity* containing a business significance to its *business partners* are discovered. They contain a lifecycle that undergo changes of state during the execution of a *business process*. The internal change of state occurs in stages and are known as *business entity states*. Based on the importance of the business entity lifecycle, a business entity lifecycle is defined for the *business entity*. For example, a transaction amount might subject to change during an entire business process activity however a lifecycle is not required to be created.

Table A.5 in the appendix, shows the *business entity* worksheet template and Table 5.5 shows the information entered in the worksheet for the *business entity* `Quote`. This worksheet provides a decription of the *business entity*, information registered in the business library, captures the requirements for the *business entity* lifecycle gathered during the interview between a business domain expert and a business analyst. A description of the business entity lifecycle is stored as well as its pre- and post-conditions and begin/end characteristics. Descriptions for every *business entity state* are also gathered here.

Business Entity	
General	
Business Entity Name	Quote
Description	Request and provide the most current prices and quantities at which the required products can be bought or sold.
Business Library Information	
URI	
BusinessTerm	Quotation; Bid; Tender; stated price
Version	1.0
Status	Approved
Owner	UN/CEFACT
Copyright	UN/CEFACT
Reference(s)	
Lifecycle	
Pre-Condition	None
Post-Condition	If the quote is accepted then an order is placed. If rejected the customer is notified.
Begins When	A price of quotes for required goods is requested by the buyer.
Ends When	The quote is accepted or rejected
Exceptions	None
Lifecycle States (add more Business Entity States if needed)	
Business Entity State	
Name	Requested
Description	The buyer sends a request for the prices of required goods to the seller
Preceding State(s) including events and transition conditions	None
Valid Actions	None
Business Entity State	
Name	Processed
Description	The requested list of prices for the products are updated by the seller.
Preceding State(s) including events and transition conditions	requested
Valid Actions	The requested price for products are either accepted or rejected by the seller.

Business Entity State	
Name	Provided
Description	If the customer is in the accepted list of the seller, the quote is provided.
Preceding State(s) including events and transition conditions	Processed
Valid Actions	A detailed list of ordered quote products is provided to the customer
Business Entity State	
Name	Refused
Description	If the customer is found in the sanction list, the quote is rejected.
Preceding State(s) including events and transition conditions	Processed
Valid Actions	The customer is notified

Table 5.5: Business Entity Worksheet - Quote

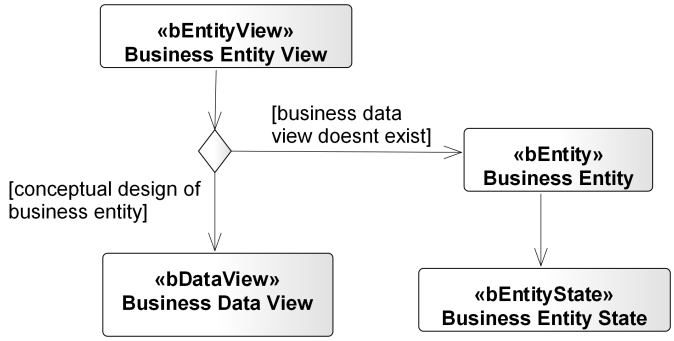

Figure 5.17: Workflow - Business Entity View

5.4.3.1.1 Step-by-Step modeling guide

Step 1. ***Identify and create business entities for the business entity view.***

Overview and purpose

A *business entity* is a real world object (e.g. order, invoice) having business significance to two or more *business partners*. It can contain a lifecycle model describing a series of changes, through which the *business entity* evolves from

its original stage e.g. processing an order. *Business entities* are identified during the execution of *business processes* where its lifecycle represents the *business process* milestones. However, *business entities* for *business processes* need not be mandatorily identified.

Model business entities
A *business entity* is a class stereotyped as *BusinessEntity (bEntity)* created beneath the *business entity view*. At least one *business entity* needs to be defined in the *business entity view*.

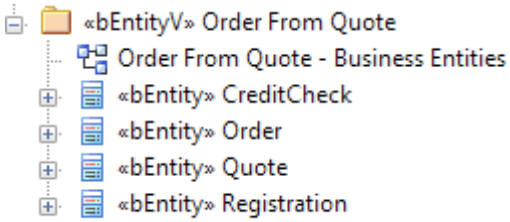

Figure 5.18: Business Entity View - Order From Quote

Figure 5.18 shows a tree view representation of *business entity*: `CreditCheck, Order, Quote, Registration` which are discovered and created beneath the *business entity view* `OrderFromQuote`.

Step 2. ***Identify and create business data views for the business entity view.***

Overview and purpose
A *business data view* is used to describe the first conceptual design of a *business entity*. This conceptual design is used to create a business document of the *business entity* used in a later stage of the UMM modeling process. *Business data views* are discovered while modeling *business processes* and are found essential before designing a *business entity* lifecycle.

Model a business data view
A *business data view* is a package stereotyped as *BusinessDataView (bDataV)*and created beneath the *business entity view*. The conceptual design of the *business entity* is represented in the form of a UML Class Diagram within a *business data view*. A *business data view* is optional within the *business entity view*. At least one class describing the *business entity* needs to be defined within the UML class diagram of a *business data view* when created.

Step 3. ***Create business entity states for a business entity.***

Overview and purpose
A business entity lifecycle reflects the milestones and business objectives achieved by the *business entity* during the execution of a *business process*. It is described as a flow of *business entity states*.

Describe the lifecycle of a business entity.
Business entity state stereotyped as *BusinessEntityState (bEState)* is created beneath the *business entity view*. A *business entity* can contain any number of *business entity states* or no *business entity states* at all. A UML state machine diagram is found most appropriate for designing *business entity states* of a *business entity*.

Business entity states of a *business entity* can be either internal or shared *business entity states* in a *business process*. More information on this is demonstrated in section 5.4.1, step-by-step modeling guide - step 6. A *business entity state* can also be used to represent a precondition or post condition to begin or end a *business process* and is noted down as a constraint in its respective *business process use case*.

Taking the *business entity*, `Quote` into consideration, Figure 5.19 describes the lifecycle starting with the initial state. Once the *buying organization* has requested the `Request Price Quote`, It reaches the `requested` state. The business entity state `processed` is set when the quote is being examined by the *selling organization*. If the quote is accepted by the *selling organization*, then the `Quote` is provided to the *buying organization* setting the business entity state to `provided`. If the customer is in the sanction list then the customer is notified that the requested `Quote` is refused, setting the *business entity state* of the entity `Quote` to `refused`.

5.4.3.2 Tagged Values of stereotypes within the *bEntityView*

The *bDataV* contains the following tagged values:

- *uniqueidentifier*: A unique identifier represents the *business library* package in the registry. A UUID (Universally Unique Identifier) is used within the URI structure scheme to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library.*
- *versionidentifier*: Version identifier is the current version of the *business library* in the registry. The version information should not be assigned by the modeler rather must be managed in the registry. An unregistered library must not have any version assigned.

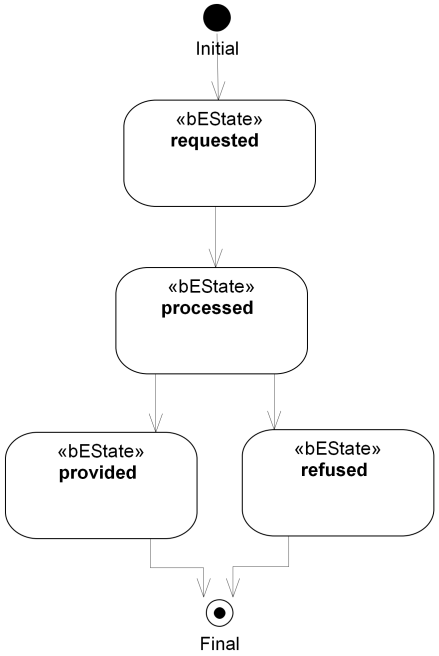

Figure 5.19: Business Entity State Machine Diagram - Quote

- *businessTerm*: A synonym which is frequently used by the *business data view* in business.

- *copyright*: Copyright of the *business library* package.

- *owner*: The owner of the *business library* which could be an organization, institution or an individual.

- *reference*: Contains location information to additional resources where further information about the *business library* could be found.

- *status*: It represents the current lifecycle status of the *business data view* registered in the *business library* package. The status is managed in the registry. The possible status representing the *business data view* are *proposed, approved, mandatory, validated* and *implemented*. An registry unregistered must not have any status information assigned.

The stereotypes *bEntity* and *bEState* do not have any predefined tagged values.

5.4.3.3 Summary of artifacts within the *bEntityView*

All real world things that are of business significance to the business processes are discovered in the *business entity view*. Results of the artifacts created in this include:

- ***business entity***: A class identified during the execution of *business processes* and is a real world object. In a collaborative *business process* it is shared among two or more business partner types.
- ***business entity state***: A state representing a specific state of a *business entity* that changes during the execution of a *business process*.
- ***business data view***: A container describing the conceptual design of a *business entity*.

5.4.0.4 Summary of artifacts created within the *bRequirementsView*

The *bRequirementsView* results in the following artifacts:

- The ***business domain view*** is a package which is used to identify and model the workflow of *business processes* of a business domain. Based on the classification schema, *business processes* are discovered and classified.
- The ***business partner view*** is a package that contains a list of *business partners* and *stakeholders* of a business domain.
- The ***business entity view*** is a package that contains *business entities* describing their lifecycle through *business entity states* having business significance in the business domain.

5.5 Business Choreography View

5.5.0.1 Overview and purpose

After the flow of business activities have been visualized in the *business requirements view*, defining the order of business information exchange between organizations from a global perspective is the next phase of the UMM modeling process. A business is initially organized on how the internal business processes are executed in order to achieve its business goal. However, this influence is of more importance as soon as processes interact with other processes outside the business. In such a situation, *business partners* need to discuss and agree upon a way to collaborate. UMM uses the *business choreography view* to describe multi-party collaborations. The main use of the choreography description is to precisely define the order of business interactions between organizations or processes from a global perspective in order to promote a common understanding between participants.

5.5.0.2 Step-by-Step modeling guide

The *business choreography view* is built on three sub views that define the flow of information exchange between two or more independent organizations and describe how they should cooperate.

1. Business Transaction View
2. Business Collaboration View
3. Business Realization View

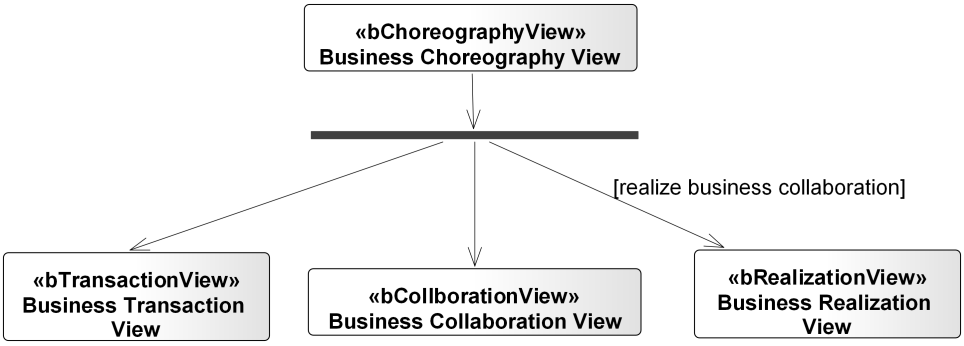

Figure 5.20: Workflow - Business Choreography View

The *business transaction view*, a package stereotyped as BusinesTransactionView *(bTransactionV)* describes the flow of binary information exchange between exactly two participants who play a certain role as they perform business activities. The *business collaboration view* is a package stereotyped as BusinessCollaborationView *(bCollaborationV)* which describes the choreography of multiple *business transactions*. Finally, the *business realization view* is a package stereotyped as BusinessRealizationView *(bRealizationV)* that describes the execution of a business collaboration between different sets of participants.

As shown in Figure 5.21, the *bChoreographyView* `Order From Quote` contains three *business transaction views*: `Register Customer, Request For Quote` and `Place Order`, two *business collaboration views*: `Register Customer Collaboration`, `Order From Quote Collaboration` and one *business realization view*: `Order From Quote`.

5.5.0.2.1 Tagged Values of stereotypes within the *bChoreographyV*

The stereotypes *bTransactionV*, *bCollaborationV* and *bRealizationV* must contain the following tagged values:

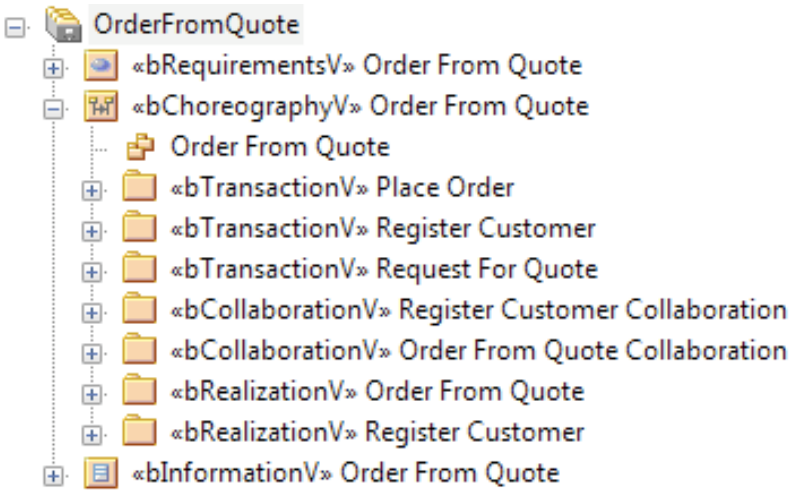

Figure 5.21: Business Choreography View - Order From Quote

- *uniqueidentifier*: A unique identifier represents the *business library* package in the registry. A UUID (Universally Unique Identifier) is used within the URI structure scheme to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library.*

- *versionidentifier*: A version identifier is the current version of the *business library* in the registry. The version information should not be assigned by the modeler rather must be managed in the registry. An unregistered library must not have any version assigned.

- *businessTerm*: A synonym frequently used by the stereotype in business.

- *copyright*: Copyright of the *business library* package.

- *owner*: The owner of the *business library* which could be an organization, institution or an individual.

- *reference*: Contains location information to additional resources where further information about the *business library* could be found.

- *status*: It represents the current lifecycle status of the stereotype registered in the *business library* package. The status is managed in the registry. The possible status representing the *business choreography view* are *proposed, approved, mandatory, validated* and *implemented*. An registry unregistered must not have any status information assigned.

5.5.1 Business Transaction View

5.5.1.1 Overview and purpose

When enterprises follow a path towards interacting across and between organizations, integrated *business processes* become increasingly complex. Basic interactions between organizations are defined as transactions. In the *business transaction view*, business messages exchanged between two organizations are known as *business transactions*. The purpose of the *business transaction view* is to model the structure of business information exchanged between organizations. In this view, a business analyst models the *business transaction view* based on the knowledge captured from the business experts in the business requirements view. A *business transaction* synchronizes the state on both side of the participants by resulting in a state of change, of a *business entity*. The *shared business entity state* is used to coordinate the state of change on both sides of information systems.

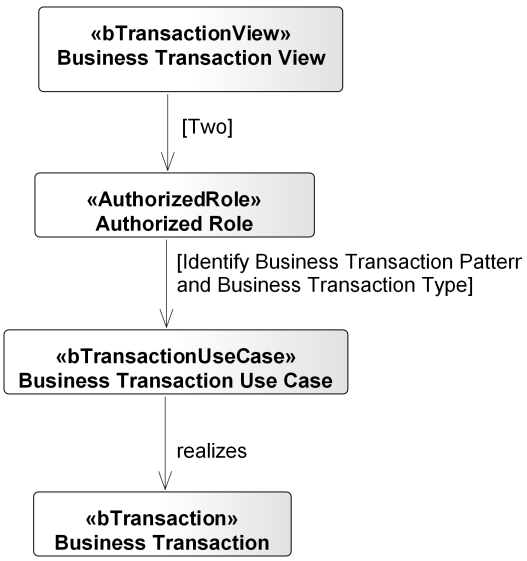

Figure 5.22: Workflow - Business Transaction View

5.5.1.2 Step-by-Step modeling guide

Step 1. ***Identify and create authorized roles for a business transaction.***

Overview and purpose of authorized roles
Initially, a modeler should identify the parties which participate in a *business*

transaction. UMM distinguishes between the concept of *authorized roles* (e.g. requestor) in *business transactions* and *business partners* (e.g. Purchasing Organization) in *business processes* (*refer section 5.4.2*). The main reason behind this is because *business transactions* might be reused between different sets of *business partners* in different scenarios of a same business domain or different domain. *Authorized roles* are only valid in the namespace they are defined in. If an *authorized role* identifies an event that changes the state of a *shared business entity*; *refer Section 5.4.1, step-by-step modeling guide - step 6*, it initiates a *business transaction*. The *business transaction* is responsible for keeping the all the relevant *business entities* in the same state in both information systems.

Model authorized roles

Authorized roles are actors stereotyped as *AuthorizedRoles* created beneath the *business transaction view*. Exactly two *authorized roles* must be defined within a *business transaction view*, since only two participants can take part in a *business transaction*. If an *authorized role* participates in more than one *business transaction* it needs to be uniquely defined in its own *business transaction view*.

Figure 5.23 shows two *authorized roles* `Buyer` and `Seller` modeled for participating in the *business transaction* `Place Order`. The `Buyer` plays the role of placing an order and the `Seller` plays the role of accepting or rejecting the requested order.

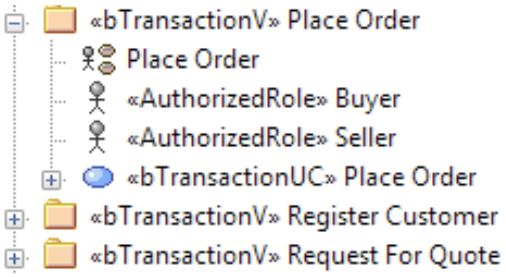

Figure 5.23: Business Transaction View - Place Order

Step 2. ***Identify the business transaction type.***

A *business transaction* synchronizes the state of both information systems. Hence, in order to design a *business transaction* we first need to distinguish the types of *business transactions*. UMM differentiates between two major types of *business transactions*: one-way and two-way. In a one-way *business transaction*, business information flows only from the initiating *authorized role* (requestor, initiator) to the responding *authorized role* (responder). The requestor

results in an already effective and irreversible state change that the responder has to accept. For e.g., notification of a shipment or an update of a product in a catalogue.

In a two-way *business transaction*, the business information flows from the initiator to the responder and sets the *business entity* to a provisional state. This business information is sent back from the responder to the initiator to set the final and irreversible state change. In a business context, irreversible means returning to an original state requires another compensating *business transaction*. For e.g., once a purchase order is agreed upon in a *business transaction*, a rollback is not allowed any more but requires the execution of a cancel order *business transaction*.

In our example model, `Place Order` is distinguished as a two-way *business transaction* type.

Step 3. ***Determine the business transaction pattern.***

A basic pattern for *business transactions* realized by the design technique of analogy is provided and has to be used when creating a new business transaction. In UMM, we categorize two, one-way business transaction patterns [8]: *Notification* and *Information Distribution* and four, two way business transaction patterns: *Query/Response, Request/Response, Request/Confirm, Commercial Transaction*. Based on the business transaction pattern that is initialized, a business transaction follows the same pattern until it completes its execution process.

Notification (one-way): This business pattern represents a formal, unidirectional information exchange between two *business partners*. It is applied on a *business transaction* when the requestor is notified that the responder has reached an irreversible *business entity state*. An example is a notification of shipment.

Information Distribution (one-way): This pattern represents an informal, unidirectional information exchange between two *business partners*. Receiving information about the price discounts to customers is one typical example.

Commercial Transaction (two-way): A commercial transaction is a common "offer and acceptance" business interaction between two *business partners*. Both the parties enter into a commitment or in other words, a commercial transactions results in a residual obligation to fulfill the terms of contract. An example is a submission of an order or receipt of purchase order response.

Query/Response (two-way): In this pattern the requested information is available in prior to the request at the responders end. For e.g. requesting the price

of a product. The requested information may be static information (or slow changing) and which isn't depended on the identity at the requestors end.

Request/Response (two-way): In this pattern, the requesting *business partner* makes a request which requires some business processing at the responders end before any response could be returned immediately (non static information). They follow the same requirements as that of the commercial transaction. This pattern results in no residual obligation to fulfil the terms of contract between two *business partners*. An example would be requesting a quote product. This request does not mandate the requestor to buy the quoted product. Similarly, the responder does not pledge himself to have the quoted product available in case of further orders.

Request/Confirm (two-way): In this pattern the requesting *business partner* requires a confirmation about a previously established business contract. This pattern might be used to get the status information. A typical example would be request order status information.

In our model `OrderFromQuote`, the *business transaction* `Place Order` follows a two-way commercial transaction pattern which contains a response from the responder.

Step 4. ***Discover the business transaction use case.***

The requirements for designing a *business transaction* between two *authorized roles* are gathered in a *business transaction use case*. Within the *business transaction views*, the *business transactions* discovered are classified as *business transaction use cases*. A *business transaction use case* is realized by *business transaction*.

Model the business transaction use case.
The *business transaction use case* stereotyped as *BusinessTransactionUseCase (bTransactionUC)* is described with the help of a use case diagram beneath the *business transaction view*. A *business transaction view* defines exactly one *business transaction use case* and binds the *authorized roles* involved. The *authorized roles* are defined in the exact context of the *business transaction use case*. In the use case diagram, the *authorized roles* are linked to the *bTransactionUC* via an association stereotyped as *participates*. The flow of the business activities are described in a *business transaction*.

Figure 5.24 shows the *business transaction use case* diagram of `Place Order` between two *authorized roles*: `buyer` and `seller`.

In the *Appendix-Worksheets*, Table A.6 shows the details captured by of the requesting and responding *authorized roles*, and their business activities. The

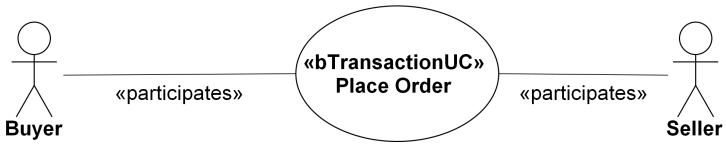

Figure 5.24: Business Transaction Use Case - Place Order

conditions and the events necessary to trigger and terminate the *business transaction* are also stated here. The *business entity states* effected due to execution of a *business transaction* are also listed here. Table 5.6 displays the worksheet for the *business transaction use case* `Place Order`. A brief description of the *business transaction*, information registered in the *business library* , details about the *business transaction* and its start and end characteristics are provided.

Business Transaction Use Case	
General	
Name	Place Order
Description	The different possibilities through which a buyer can request a list of products from the seller are described in this worksheeet. At the same time, the various means through which the seller in turn responds, for accepting or rejecting the order are described.
Business Library Information	
URI	
BusinessTerm	Request Order, Purchase Agreement
Version	1.0
Status	Approved
Owner	UN/CEFACT
Copyright	UN/CEFACT
Reference(s)	
Details	
Requesting Role	Buyer
Responding Role	Seller
Requesting Activity	Submit Order
Responding Activity	Process Order
Is Included In (Name of Business Collaboration)	Order From Quote

Start/End Characteristics	
Affected Business Entities	Order
Pre-condition	Quote is provided to the seller
Post-condition	Order is accepted; Order is rejected
Begins When	Buyer requests an order to the seller
Ends When	The buyer receives a formal acceptance or rejection of the requested order from the seller.
Exceptions	None

Table 5.6: Business Transaction Use Case Worksheet - Place Order

Step 5. ***Describe and model a business transaction.***

Overview and purpose

If an *authorized role* recognizes an event that changes the state of a *business entity*, it initiates a *business transaction* to synchronize with the collaborating *authorized role*. A *business transaction* is an atomic *business process* that involves exchanging business information between two *authorized roles*.

A *business transaction* is executed by business actions stereotyped as *BusinessAction*. *Business actions* are events performed by *authorized roles* requesting or responding to a business service and hence are defined as abstract actions. While requesting or responding to a business service, business information is exchanged between *authorized roles* through *business information envelopes*; *refer Chapter 2, Section 3.2.1*.

To initiate a *business transaction*, a requesting *business action* is executed by an *authorized role*. The requesting business information is sent from the requesting *authorized role* to the responding *authorized role* through a *requesting information pin*. The transfer of business information between *authorized roles* indicates a state of change in the *business entities*. In case of a one-way *business transaction* pattern, the change of state within a *business entity* is a final state. In case of a two-way *business transaction* pattern it could be either an interim or a final state of change in the *business entity*.

In case of a two-way *business transaction* pattern, an information pin outputs the responding business information from the responding *authorized role* to the requesting *authorized role*, in order to set a final state of one or more business entities. Furthermore, in a two-way *business transaction* pattern, if a *business transaction* is executed successfully, the resulting business document received by the responding *authorized role* needs to be evaluated. This is done with the

help of OCL constraints. OCL constraints are used to directly access the business document content or check the business document type i.e. by checking if it is a positive or a negative response from the responding business partner.

Model a business transaction.
A *business transaction* stereotyped as *BusinessTransaction (bTransaction)* is created beneath the *business transaction use case*. It is modeled as a composite activity diagram that describes the communication between the *authorized roles* and the business information exchanged between them.

The basic building blocks for building a *business transaction* are two UML partitions (swimlanes) that represent the areas of responsibility of a unique *authorized role* . Exactly two UML activity partitions that define the areas of responsibility for each *authorized roles* are stereotyped as *BusinessTransaction-Partition (bTPartition)* and created within the *business transaction* activity diagram. .These activity partitions represent two *authorized roles*; the requesting role and the responding role. The requesting role is assigned as a classifier to a requesting business swimlane and the responding role is assigned as a classifier to the responding business swimlane. Usually, the left hand side UML partition represents the initiating role and the right hand side UML partition represents the responding role. Each UML activity partition performs exactly one *business action*. The requesting UML partition executes the requesting *business action* and the responding UML partition executes the responding *business action*.

Business Action: *Business actions* are assigned to the partition of the *authorized role* executing the action. A requesting *business action* stereotyped as *RequestingBusinessAction (ReqAction)* is created within the requesting *bTPartition* of the activity diagram. A connector is used to connect the initial state to the *ReqAction*. In a similar manner, within the responding *bTPartition*, a responding *business action* stereotyped as *RespondingBusinessAction (ResAction)* is created.

Business Information Pins: The incoming or outgoing point for business information is represented by *information pins*. A requesting information pin stereotyped as *RequestingInformationPin (ReqInfPin)* is a pin connected to the *ReqAction* in the activity diagram. Similarly, a responding information pin stereotyped as *RespondingInformationPin (ResInfPin)* is a pin connected to the *ResAction* in the activity diagram.

Business Information Envelope: In case of one-way *business transaction* patterns, an envelope is added as a classifier to the *ReqInfPin* of the *ReqAction* of the requesting *bTPartition*. A transition is connected from the *ReqInfPin* of the *ReqAction* to the *ReqInfPin* of the *ResAction*. The same business information

envelope, classified to the *ReqInfPin* of the *ReqAction* is also classified to the *ReqInfPin* of the *ResAction*.

In case of two-way *business transaction* patterns, where business information needs to be transmitted from the responding *authorized role* back to the still-alive *ReqAction* of the initiating *authorized role*, there are two cases. For the following *business transaction* patterns: *request/confirm, request/response, query/response*, a *ResInfPin* is added to the *ReqAction* of the requesting and responding *bTPartition* to denote a positive reply. A transition is used to connect from the *ResInfPin* of the responding *bTPartition* to the *ResInfPin* of the requesting *bTPartition*. For the *business transaction* pattern *Commercial Transaction*, an additional *ResInfPin* is attached to the *ReqAction* of the requesting and responding UML activity partition to denote a negative reply. A transition is used to connect from the *ResInfPin* of the responding *bTPartition* to the *ResInfPin* of the requesting *bTPartition*.

Business Entity States: Considering one-way *business transactions* patterns; *notification* and *information distribution*, a *bESharedState* indicating a positive state of change is created within the requesting *bTPartition* and a connector is used to connect from the *ReqAction* to the positive *bESharedState*.

In case of two-way *business transaction* patterns, two *bESharedState* are created within the requesting *bTPartition*. One *bESharedState* is used to represent a positive state of change and another *bESharedState* is used to represent a negative state of change. Two connectors are connected from the *ReqAction* to its respective positive and negative *bESharedStates*.

Final States: In a one-way *business transaction* pattern, before any *business transaction* is concluded, two final states are added. One with with a connector from the positive *bESharedState* to a final state representing business success. In case a *business transaction* has no retries left or a time-out exception occurs where the responding *authorized role* is not able to process the message correctly, a *ReqAction* is transited to a control failure final state. Here the system issues a notification of failed business control and exits the transaction with a failure.

In case of two-way *business transaction* patterns, three final states are added within the requesting *bTPartition*. One, indicating success, second indicating failure and the third indicating control failure. A connector is used to connect the positive *bESharedState* to a final state indicating business success. Another connector, from the negative *bESharedState* to a final state, indicating failure. A final state indicating control failure is added along with a connector connected from the *ReqAction* in case of exceptions that occur during the process.

OCL Constraints: In all cases, the OCL constraints are added as condition guards on the transitions leading to its respective final state (business success or business failure). In case of a one-way *business transaction* pattern, OCL constraints are meant to be optional.

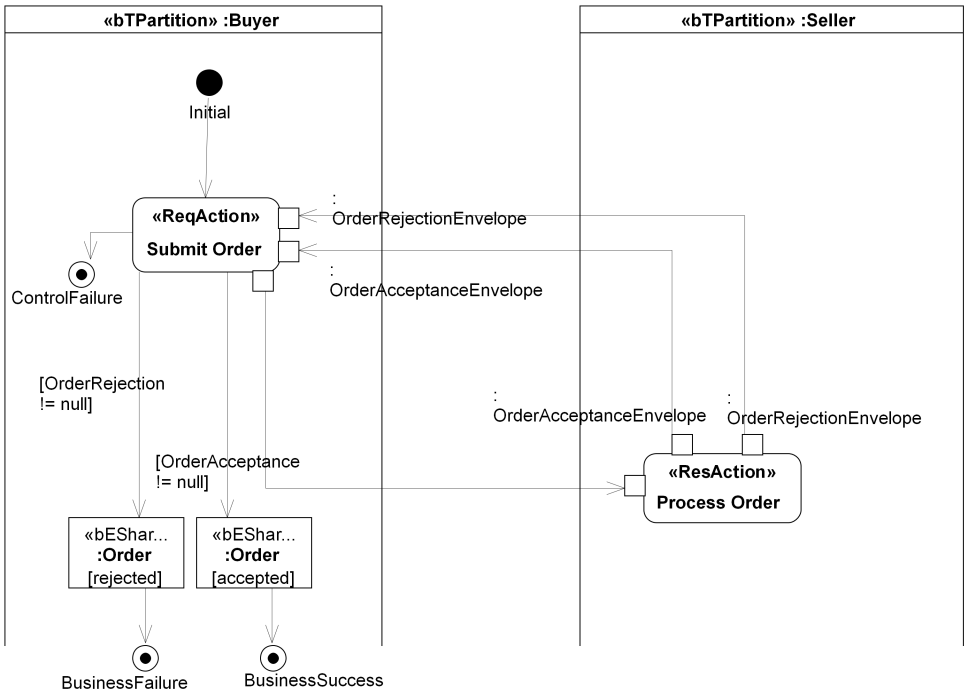

Figure 5.25: Business Transaction - Place Order

Figure 5.25 shows the activity diagram of *business transaction* `Place Order`. It consists of two *business transaction* swimlanes, one for *authorized role* `Buyer`, the other for `Seller`. The `Buyer` executes the *requesting business action* `Submit Order` setting the *business entity* `Order` to an interim state `Submitted`. The business information is transferred via *business information envelopes*. In order to set the final states of the *business entity*, the `Seller` executes the *responding business action* `Process Order` setting the *business entity* `Order` to an final state `Accepted` or `Rejected`. The input/output points for the business information exchange between *authorized roles* are shown in the diagram. Similarly, the OCL constraints are also shown representing the condition guards based on the information envelope received from the `Seller`.

In the *Appendix-Worksheets*, Table A.7 shows the business transaction worksheet template and Table 5.7 shows the worksheet for the *business transaction* `Place Order`.

Business Transaction	
General	
Name	Place Order
Description	This *business transaction* describes the pattern through which the buyer requests a list of products from the seller and at the same time, different ways through which the seller in turn, responds by accepting or rejecting the order.
Business Library Information	
URI	
BusinessTerm	Request Order; Purchase Agreement
Version	1.0
Status	Approved
Owner	UN/CEFACT
Copyright	UN/CEFACT
Reference(s)	
Details	
Select Business Transaction Pattern	Request/Response
Secure Transport	True
Requestor's Side	
Requesting Role	QuoteRequestor
Requesting Business Action Name	Obtain Quote
Time to Respond	4h
Time to Acknowledge Processing	Null
Authorization Required	False
Non Repudiation Required	False
Non Repudiation of Receipt Required	False
Intelligible Check Required	True
Number of Retries	3
Responder's Side	
Responding Role	Quote Responder
Responding Business Action Name	Calculate Quote
Time to Acknowledge Receipt	Null
Time to Acknowledge Processing	Null
Authorization Required	False
Non Repudiation Required	False
Non Repudiation of Receipt Required	False
Intelligible Check Required	True
Business Information Envelopes	
Requesting Information Envelope	
Name	Quote Request Envelope
Are Contents Confidential?	Yes
Is the Envelope Tamperproof?	Yes
Authentication Required?	Yes

Responding Information Envelope	
Name	QuoteEnvelope
Resulting Business Entity State (including transition condition)	Provided - Quote price is provided to the customer
Are Contents Confidential?	Yes
Is the Envelope Tamperproof?	Yes
Authentication Required?	Yes
Responding Information Envelope	
Name	QuoteEnvelope
Resulting Business Entity State (including transition condition)	Refused - Quote price is not provided to the customer
Are Contents Confidential?	Yes
Is the Envelope Tamperproof?	Yes
Authentication Required?	Yes

Table 5.7: Business Transaction - Place Order

5.5.1.3 Tagged Values of stereotypes within the *bTransactionV*

The stereotype *bTransactionUC* generalizes from the stereotype *bProcessUC* of the *bDomainView* thereby inheriting all its tagged values. It contains the following attributes

- *definition*: The value that needs to be created to all the participants of the *business transaction use case* is described here. The same value should already be created for its respective *business process use case*.
- *Pre-condition*: A set of conditions that need to be fulfilled before the execution of the *business transaction* and already specified in its respective *business process use case*.
- *Post-condition*: A set of conditions that need to be fulfilled after the execution of the *business transaction* and should already be specified in its respective *business process use case*.
- *Begins When*: Specifies certain business events which trigger the initiation of the *business transaction*. It also specifies a certain state that the *business process* needs to reach to initiate the *business transaction*. This should be the same values as specified in the *business process use case*.
- *Ends When*: Specifies certain business events that lead to the termination of the *business transaction* or a certain state which terminates the transaction. These events are the same as specified in the *business process use case*.

- *actions*: Describes the activities that need to be executed during the execution of a *business transaction*. These activities are the same as specified in the *business process use case* worksheet.

- *Exceptions*: Identifies the errors that occur during the execution of the *business transaction*

The stereotype *bTransaction* contains the following tagged values:

- *businessTransactionType*: An enumerated list with one of the *business transaction* patterns: notification, commercial transaction, request/response, query/response, information distribution, request/confirm.

- *isSecureTransferRequired*: This value is set as true or false based on if the business information is exchanged via a secure transport channel. The security channel ensures that the business document content is protected against unauthorized disclosure or modification and the business services are protected against unauthorized roles. This feature applies as long as the document is in the network. Once inside the enterprise, this feature does not apply. The secure transport channel must fulfil the following conditions

  - *Authenticate sender identity*: Confirm the identity of the sending role who initiates the interaction
  - *Authenticate receiver identity*: Confirm the identity of the receiving role who receives the interaction
  - *Verify content integrity*: Confirm the stability of the content exchanged during the interaction
  - *Maintain content confidentiality*: The intended receiving role only can read the content of the interaction. The information transmitted during the interaction must be encrypted when sent and decrypted when received.

The abstract stereotype *BusinessAction* contains the following tagged values:

- *Time to acknowledge receipt:* Specifies the period of time required by the receiver to acknowledge the receipt of business information sent from the sender. If the receipt of business information is not verified within the specified period of time then the initiator much retry the *business transaction*, if necessary or must send a notification of the failed *business transaction*. The initiator must exit the transaction if it is not found to verify the proper receipt of business information within the agreed period of time The period of time is specified by the senders business action. If no specific time period is required then the value here is set to null.

- *Time to acknowledge processing:* The duration of time, from when the sender sends the business information to the receiver until the time an acknowledgement of processing is received back by the sender. The need for an acknowledgement of processing takes place after the requesting business information passes a set of business rules and is handed over to the application for processing. The initiator must exit the *business transaction* if the business information sent is not acknowledged within the maximum period of time allotted. The initiator must retry the *business transaction* if needed or must send a notification of failed business input/output information if the receiver does not acknowledge processing of business information within the valid duration of time. The value is set to null if not specific time duration needs to be allotted.

- *Is authorization required:* Is set as true if the *authorized role* must authorize itself. The receiver must sign the business document exchanged and sender must validate the business authorization and approve with the receiver. The sender must output an authorization exception if the receiver is not authorized to perform the business activity.

- *Is non-repudiation required:* True, if the *authorized role* need not repudiate the execution of the business action which inputs and outputs business information.

- *Is non-repudiation of receipt required:* Is set as true, if an acknowledgement of receipt is required from the receiver. It ensures that the business information received by the receiver sends a signed receipt. It also indicates that the responding party must not be able to repudiate the execution of sending the signed receipt.

- *Is intelligible check required:* Is set as true if the receiver must check that the business information is not garbled during transmission before an acknowledgement of receipt is sent back to the sender. Verification of receipt is returned only when the documents content is readable.

The stereotype *ReqAction* contains additionally two more tagged values:

- *Time to respond:* Specifies the period of time required by the responding role to reply in a two-way *business transaction*. In case of a one-way *business transaction*, the value is set to null.

  The requesting *authorized role* must exit the transaction if the responding business role is not able to return the responding business information within the agreed period of time. The requesting *authorized role* must retry a *business transaction* if necessary or must send notification of failed business control if the responding *authorized role* does not deliver the responding business information within the agreed amount of time period.

- *Retry count:* Specifies the number of times, the requesting *authorized role* to re-initiate a *business transaction* in case of a time-out exception like exceeding the *time to acknowledge receipt, time to acknowledge processing* or the *time to respond*. The *retry count* does not include the first attempt and it does not cover the exceptions occurred due to error document content or bad message sequence.

The abstract stereotype *InfPin* contains the following tagged values. Stereotypes *ReqInfPin* and *ResInfPin* generalizes from *InfPin*, thus inheriting all the tagged values from *InfPin*.

- *isAuthenticated*: The flag set as true, if the sender's digital certificate is linked with the document entity. This signifies that a proof of the signer's identity is required.

- *isConfidential*: The flag is set as true, if the exchanged information is encrypted so that third parties are not authorized to read the content.

- *isTamperProof*: The flag is set as true, if the information that is exchanged between the requestor and responder has an encrypted message digest associated with it, in order to check if the message has be tampered. This is done with the help of a digital signature associated with the document entity. The digital certificate is nothing else other than the senders digital certificate and the encrypted message digest.

In case if *intelligible required* is set per default as true, every business document exchanged during a *business transaction* needs to be reviewed as to if it is readable by the receiver before sending an acknowledgement of receipt.

Table 5.8 shows the default tagged values for the *requesting business action* for all *business transaction* patterns. These tagged values are the requirements, the responder needs to be fulfill in order the intended *business transaction* pattern takes place successfully.

Tables 5.9 shows the default assignment of tagged values for the *responding business action* for all *business transaction* patterns. These tagged values are the requirements, the requestor needs to be fulfill in order the intended *business transaction* pattern takes place successfully.

5.5.1.4 Summary of artifacts created within the *bTransactionV*

Results of the artifacts created within the *business transaction view* include the following:

- ***authorized roles*** : Actors participating in a *business transaction*. It is used as a more generic term than *business partners* participating in business processes.

	Time to Acknowledge Receipt	Time To Acknowledge Processing	Time To Respond	Is Authorization Required	Is Non-Repudiation Required	Is Non-repudiation of receipt required	Retry Count	Is Intelligible Check Required
Commercial Transaction	2h	6h	24h	True	True	True	3	True
Request/Confirm	Null	Null	24h	False	False	False	3	True
Request/Response	Null	Null	4h	False	False	False	3	True
Query/Response	Null	Null	4h	False	False	False	3	True
Notification	24h	Null	Null	True	True	True	3	True
Information Distribution	Null	Null	Null	False	False	False	0	True

Table 5.8: Default Tagged Values - Requesting Role

	Time to Acknowledge Reciept	Time To Acknowledge Processing	Is Authorization Required	Is Non-Repudiation Required	Is Non-repudiation of receipt required	Is Intelligible Check Required
Commercial Transaction	2h	6h	True	True	True	True
Request/Confirm	2h	Null	True	False	True	True
Request/Response	Null	Null	False	False	False	True
Query/Response	Null	Null	False	False	False	True
Notification	Null	Null	False	False	False	True
Information Distribution	Null	Null	False	False	False	True

Table 5.9: Default Tagged Values - Responding Role

- ***business transaction use case*** : A use case that extends the *business process use case* and captures the requirements of a *business transaction* between exactly two authorized roles. It cannot be further refined. *Authorized roles* participate in the *business transaction use case*.

- ***participates***: An association between an *authorized role* and a *business transaction use case*. This association indicates that an input or output of information is provided by the *authorized role* to its associated *business transaction*.

- ***business transaction***: An activity that describes the interactions between *authorized roles*. There are two types of *business transactions*. One-way *business transaction*, which results in an irreversible state change of a business object. Two-way *business transaction* which results in an interim state requiring another call of a *business transaction*, to set the final and irreversible state change of the business object We distinguish between 2, one-way *business transactions* and 4, two-way *business transactions*.

- ***business transaction partition***: A partition allocated for an *authorized role*. All the business activities of a *authorized role* are performed within the business activity partition.

- ***business action***: An action executed by the *authorized role* in a *business transaction*. It is an abstract stereotype which could be either a *requesting business action* or a *responding business action*.

- ***Requesting business action***: An action executed by an *authorized role* requesting a business service from another *authorized role*.

- ***Responding business action***: An action executed by an *authorized role* responding to a requested business service from another *authorized role*.

- ***Business Information pin***: An abstract concept representing the output/input point of sending and receiving business information between *authorized roles*.

- ***Requesting business information pin***: A pin containing business information that needs to be transferred from the requesting *authorized role* to the responding *authorized role* resulting in an irreversible state of change in one-way *business transaction* and interim state in case of two-way *business transaction*.

- ***Responding business information pin***: A pin containing business information that needs to be transferred from the responding *authorized role* to the requesting *authorized role*, only in case of a two-way *business transaction* pattern .

5.5.2 Business Collaboration View

5.5.2.1 Overview and purpose

Business transactions are developed and constructed based on different business interactions. One of the main objective in UMM is to encourage re-use of existing *business transactions*. Multiple *business transactions* can be linked and composed to perform a common business objective also known as *business collaboration*. *Business collaborations* coordinate the flow of business information across organizations and link their *business processes* in order to ensure a consistent outcome. This means that enterprises must embrace the benefits of cooperating with one another. When a larger goal needs to be achieved, enterprises can benefit from working together and achieve the most effective solutions. In a *business domain view*, a *business process* is a set of activities aiming to accomplish a certain goal to a *business partner*. In the *business choreography view*, *business collaboration* is a special kind of *business process* characterized by the fact that the business activities are executed by two or more *business partners* who play a certain role in order to achieve their goal.

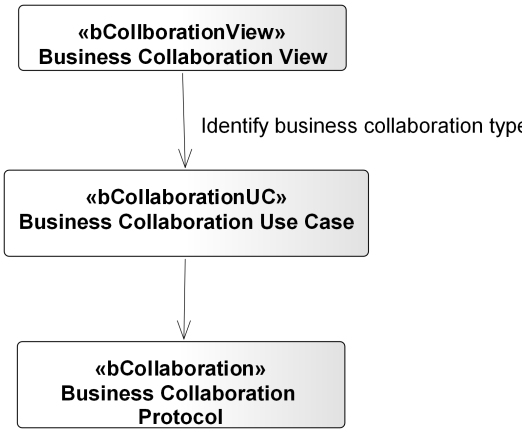

Figure 5.26: Workflow - Business Collaboration View

5.5.2.2 Step-by-Step modeling Guide

Step 1. ***Identify the type of*** **business collaboration.**

There are two kinds of business collaborations.

- Binary collaboration, where only two *authorized roles* participate. A binary collaboration is also known as a *business transaction*. A *business*

transaction is a special kind of *business collaboration*. It is the basic building block for designing complex *business collaborations*.

- Multi party collaboration involves more than two *authorized roles*. This type of *business collaboration* is built on multiple *business transactions* and the *business collaboration* defines an execution order of the *business transaction*. A *business collaboration* is used to combine multiple *business transactions* and/or *business collaborations*. Each of the *business transaction* or collaboration is defined in its own *business transaction view* and *business collaboration view* respectively. Before designing the choreography for complex *business transactions*, the requirements are collected in a *business collaboration use case*.

Step 2. ***Identify and describe the business collaboration use case.***

Overview and purpose
A *business collaboration use case* captures the requirements for designing the choreography of multiple *business transactions* or/and *business collaborations* just as a *business transaction use case* captures business requirements for designing a business interaction between exactly two *authorized roles*.

Model a business collaboration use case.
A *business collaboration use case* stereotyped as *bCollaborationUC (BusinessCollaborationUseCase)* is created beneath the *business collaboration view* and represented as a use case diagram. Exactly one *business collaboration use case* is defined within the *business collaboration view* since a business choreography of exactly one business collaboration can be described.

The *business collaboration use case* aggregates multiple *business transaction use cases* or recursively structured *business collaboration use cases*. This is manifested by *include* dependency in the use case diagram. The important goal of UMM is to enable re-use. Hence a *business transaction use case* must be included in at least one *business collaboration use case*. In other words, it is quite necessary that a *business collaboration use case* includes at least one *business collaboration use* case or a *business transaction use case*.

At least two *authorized roles* must take part in a *business collaboration use case* and hence they should be defined within the context and namespace of the *business collaboration view*. If an *authorized role* participates in multiple business collaborations performing the same role then it needs to be defined with the same name within its respective *business collaboration view*. The *authorized roles* are associated to the *business collaboration use case* in the use case diagram using an association stereotyped as *participates*.

In order to enable re-use of existing *business transactions* in different *business collaborations*, the use of different names for *authorized roles* in *business transactions* are recommended. In other words, when a *business transaction* is used in two different *business collaboration use cases*, it must be used with a different set of names for *authorized roles*. These *authorized roles* however, should already be defined in a *business transaction view* or another *business collaboration view* which is included in the current *business collaboration view*.

A business collaboration can be extended by multiple *business collaboration use cases*. A *business collaboration use case* can optionally extend zero to multiple *business collaboration use cases* and are linked through an *extends* dependency .

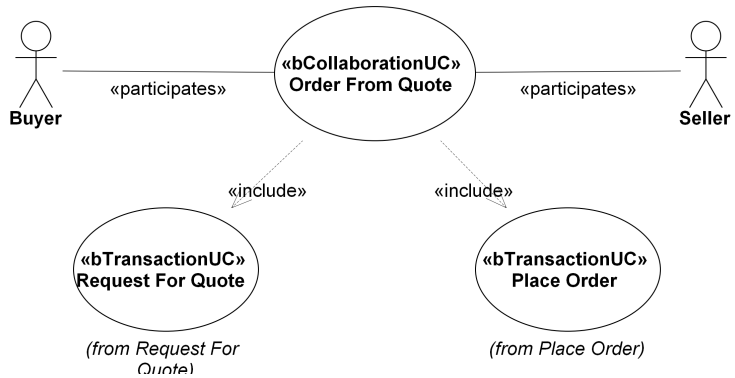

Figure 5.27: Order From Quote- Business Collaboration Use Case

Figure 5.27 shows two *authorized roles* `Buyer` and `Seller` participating in the *business collaboration use case* `Order From Quote`. The *business transaction use cases* `Request For Quote` and `Place Order` are aggregated in this *business collaboration use case* through an *include* dependency.

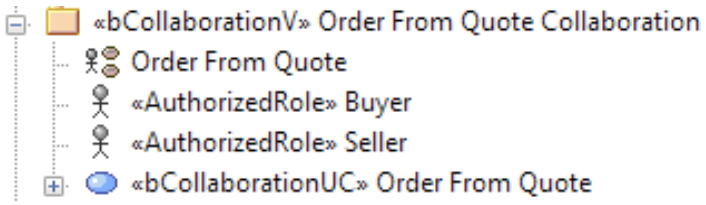

Figure 5.28: Business Collaboration Use Case - Tree View

Figure 5.28 shows a tree view representation of the same *business collaboration use case* `Order From Quote`.

In the *Appendix-Worksheets*, Table A.8 shows the *business collaboration use case* worksheet template and Table 5.10 represents the worksheet for the *business collaboration use case* `Order From Quote`.

Business Collaboration Use Case	
General	
Name	Order From Quote
Description	This business collaboration provides a quote requestor, the possibilities to request a quote for the required items and provides a quote responder with different possibilities to provide a formal quote or reject the quote. In case a quote is provided, the buyer is provided with the means to send the seller a list of items they wish to purchase and the seller is provided with the means to send the buyer a formal acceptance or rejection of the order placed.
Business Library Information	
URI	
BusinessTerm	Request Order; Purchase Agreement
Version	1.0
Status	Approved
Owner	UN/CEFACT
Copyright	UN/CEFACT
Reference(s)	
Participants	
Participating Role	Buyer; Seller
Is Included In (Name of parent Business Collaboration – if there is any)	None
Start/End Characteristics	
Affected Business Entities	Quote; Order
Pre-condition	Customer is registered.
Post-condition	Order is accepted or rejected
Begins When	A quote requestor sends a list of items to the quote responder for quotation.
Ends When	The buyer receives a formal acceptance or rejection of the order that is placed from the seller.
Exceptions	None
Included Business Transaction Use Cases	
Business Transaction Use Case Name	Request For Quote; Place Order

Table 5.10: Business Collaboration Use Case - Order From Quote

Step 3. ***Describe the business collaboration protocol.***

Overview and purpose

A *business collaboration protocol* describes the collaborative *business process* existing between two or more *authorized roles*. This protocol is based on the requirements gathered from its corresponding *business collaboration use case*. It is a complex *business transaction* that does not meet the basic principles of transactions and should be used where transaction rollback is not possible. The collaborative behaviour of a *business collaboration protocol* is described through *business transaction calls*, *business collaboration calls* and *nested business collaborations*.

A *business transaction call* is used to call a predefined *business transaction* from a *business transaction view*. A *business collaboration call* is used to execute a unique *business collaboration protocol* from its respective *business collaboration view*. During the execution process of the *business transaction call*, before any response is sent back to the requesting *authorized role*, more *business partners* might need to be involved during the execution process. In this case, a *nested business collaboration* is used.

In order to define the areas of responsibility for each *authorized role*, the concept of *business collaboration partitions* are used. A *business collaboration partition* represents a unique *authorized role*. Modeling the choreography of a *business collaboration protocol* focuses not only on the information exchange but also on the coordination and communicative aspects of interactions. This is fulfilled with the help of role mapping by designing the information flow between business collaboration partitions and *business transaction/collaboration calls* and/or *nested business collaborations*.

Model a business collaboration protocol

A *business collaboration protocol* stereotyped as *bCollaborationProtocol (BusinessCollaborationProtocol)* is created beneath the *business collaboration use case* and is represented as a composite UML activity diagram. Exactly one *business collaboration protocol* can be defined within a *business collaboration use case*.

Business collaboration partitions are UML activity partitions stereotyped as *BusinessCollaborationPartition (bCPartition)* and modeled in a UML activity diagram of the *business collaboration protocol*. Each *business collaboration partition* represents a unique *authorized role* which is assigned as a classifier and vice versa a unique *authorized role* is assigned to a *business collaboration partition*. A partition serves the purpose of role mapping.

At least one *business transaction call* or *business collaboration call* must exist within a *business collaboration protocol*. An initial state is created within the activity diagram. A transition is created from the initial state to a *business collaboration call/ business transaction call*.

In a *business collaboration protocol*, the *authorized roles* are not classified as initiator and responder as in the *business transaction* since a *business collaboration partition* can represent a initiator for one *business transaction/ collaboration call* at the same time a responder for another *business transaction/collaboration call*.

Business Transaction Call: A *business transaction call* is a UML action stereotyped as *bTransactionAction (BusinessTransactionCall)* and created within a *business collaboration partition* of a *business collaboration protocol* . A *business transaction call* which in turn calls its corresponding *business transaction use case* is initiated by an *authorized role* from the *business collaboration protocol*. A *business transaction* must be called by at least one or multiple *business transaction calls* from different *business collaboration protocols*. A *business collaboration protocol* can be composed of zero to multiple *business transaction calls*.

The role mapping for a *business transaction call* is designed as follows:

- *Requesting information flow between business collaboration partitions and business transaction call*

  When a *business transaction call* is executed between two *authorized roles* of a *business collaboration protocol*, two steps are required to model the *requesting information flow* stereotyped as *initFlow* between a *business transaction call* and its *business collaboration partition* respectively.

  Firstly, a role mapping is defined through an *initFlow* from an *authorized role* of a *business collaboration partition* to the *business transaction call* which executes a *business transaction* containing an initiating *authorized role*. The *business collaboration partition* classified by the *authorized role* is the source of the initiating flow and the *business transaction call* is the target.

  Secondly, a role mapping is defined from the responding *authorized role* of the same *business transaction call* to a *business collaboration partition* through an information flow stereotyped as *initFlow*. In this case, the *business transaction call* becomes the source of the initiating flow and the *business collaboration partition* becomes the target.

  During the role mapping, the *authorized role* of *business collaboration partition* and the *authorized role* of the *business transaction*, called by

the *business transaction call* need not have identical names. If they have the same names, then the information flow specifically defines the role mapping between the *authorized roles*. If not, then the *authorized role* of the *business collaboration partition* is classified by the *authorized role* of the *business transaction*.

- *Responding information flow between business collaboration partitions and business transaction call.*

  The *responding information flow* is designed only in the case of a two-way *business transaction* pattern. Similar to the initiating flow, two steps are required to model the *responding information flow* stereotyped as *reFlow* between a *business transaction call* and its *business collaboration partition* respectively.

  Firstly, a role mapping is defined through an information flow stereotyped as *reFlow* from an *authorized role* of a *business collaboration partition* to the *business transaction call* which executes a *business transaction* containing also an initiating *authorized role*. The requesting *business collaboration partition* is the source of the initiating flow and the *business transaction call* is the target.

  Secondly, a role mapping is defined from the same *business transaction call* containing a responding authorized role to a *business collaboration partition* which initiated the *business transaction call* through an information flow stereotyped as *reFlow*. In this case, the *business transaction call* becomes the source of the initiating flow and the *business collaboration partition* becomes the target. With this, the responding flow of information completes the *business transaction call* at the initiators side of the *business collaboration protocol*.

  During the role mapping, the *authorized role* of *business collaboration partition* and the *authorized role* of the *business transaction*, called by the *business transaction call* need not have identical names. If they have the same names, then the information flow specifically defines the role mapping between the *authorized roles*. If not, then the *authorized roles* of the *business collaboration partition* is classified by the *authorized role* of the *business transaction*.

Business Collaboration Call : A *business collaboration call* is a UML action stereotyped as *bCollaborationAction (BusinessCollaborationCall)*. Not every *business collaboration protocol* is called by a *business collaboration call*. A *business collaboration protocol* can be called by multiple business collaboration calls. A *business collaboration protocol* can be composed of zero to many

business collaboration calls. The role mapping design for a *business collaboration call* is designed as follows:

- *Requesting information flow between business collaboration partitions and a business collaboration call*

  When a *business collaboration call* is executed between two *authorized roles* of a *business collaboration protocol*, two steps are required to model the *requesting information flow* stereotyped as *initFlow* between a *business collaboration call* and its *business collaboration partition* respectively.

  Firstly, a role mapping is defined through an information flow stereotyped as *initFlow* from an *authorized role* of a *business collaboration partition* to the *business collaboration call* which executes a business collaboration containing also an initiating authorized role. The *business collaboration partition* is the source of the initiating flow and the *business collaboration call* is the target.

  Secondly, a role mapping is defined from the responding *authorized role* of the same *business collaboration call* to a *business collaboration partition* through an information flow stereotyped as *initFlow*. In this case, the *business collaboration call* becomes the source of the initiating flow and the *business collaboration partition* becomes the target.

  During the role mapping, the *authorized role* of *business collaboration partition* and the authorized role of the business collaboration, called by the *business collaboration call* need not have identical names. If they have the same names, then the information flow specifically defines the role mapping between the *authorized roles*. If not, then the *authorized role* of the *business collaboration partition* is classified by the *authorized role* of the *business collaboration call*.

- *Responding information flow between business collaboration partitions and business collaboration call*

  A *responding information flow* is designed only in the case of a two-way *business transaction* pattern. Similar to the initiating flow, two steps are required to model the *responding information flow* stereotyped as *reFlow* between a *business collaboration call* and its *business collaboration partition* respectively.

  Firstly, a role mapping is defined through an information flow stereotyped as *reFlow* from an *authorized role* of a *business collaboration partition* to the *business collaboration call* which executes a business collaboration containing an initiating *authorized role*. The *business collaboration parti-*

tion is the source of the initiating flow and the *business collaboration call* is the target.

Secondly, a role mapping is defined from the *authorized role* of the same *business collaboration call* to a *business collaboration partition* which initiated the *business collaboration call* through an information flow stereotyped as *reFlow*. In this case, the *business collaboration call* becomes the source of the initiating flow and the *business collaboration partition* becomes the target. With this, the responding flow of information completes the *business collaboration call* at the initiators side of the *business collaboration protocol*.

During the role mapping, the *authorized role* of *business collaboration partition* and the *authorized role* of the business collaboration, called by the *business collaboration call* need not have identical names. If they have the same names, then the information flow specifically defines the role mapping between the *authorized roles*. If not, then the *authorized roles* of the *business collaboration partition* is classified by the authorized role of the business collaboration.

Nested Business Collaboration: A *nested business collaboration* is a UML action stereotyped as *NestedBCollaboration (NestedBusinessCollaboration)* created within the responding *business collaboration partition* of the *business collaboration protocol*. A *business transaction call* needs to have a two-way *business transaction* pattern for using a *nested business collaboration*. A *nested business collaboration* should already be defined as a *business collaboration use case* within another *business collaboration view*. Unlike a *business collaboration call* which exists between two *authorized roles*, the *nested business collaboration* is characterized by the fact that it resides within a *business collaboration partition*. Not every *business collaboration call* is a nested *business collaboration protocol*. A *business collaboration protocol* can be called by multiple *nested business collaboration*. However, not every *business collaboration partition* which contains a *business transaction call* must execute a *nested business collaboration*. The role mapping design for a *nested business collaboration* is designed as follows:

- *Requesting information flow from a business transaction call to a nested business collaboration.*

  Here, there is only one step required for modeling the information flow. To define a role mapping, the requesting *authorized role* of the *business transaction* initiates the *nested business collaboration* or in other words calls the business collaboration through an initiating flow stereotyped as *initFlow*. Here, the source of *initFlow* is the *business transaction call* and

the target is the *nested business collaboration* residing within the *business collaboration partition*.

- *Responding information flow from nested business collaboration to business transaction call.*

  In this case, there is only one step required for modeling the information flow, which however, applies only in the case of a two-way *business transaction* pattern. To define a role mapping, the responding *authorized role* of the *nested business collaboration* is mapped to the responding *authorized role* of *business transaction call* through an *responding information flow* stereotyped as *reFlow*. Here, the source of *reFlow* is the *nested business collaboration* and the target is the *business transaction call*.

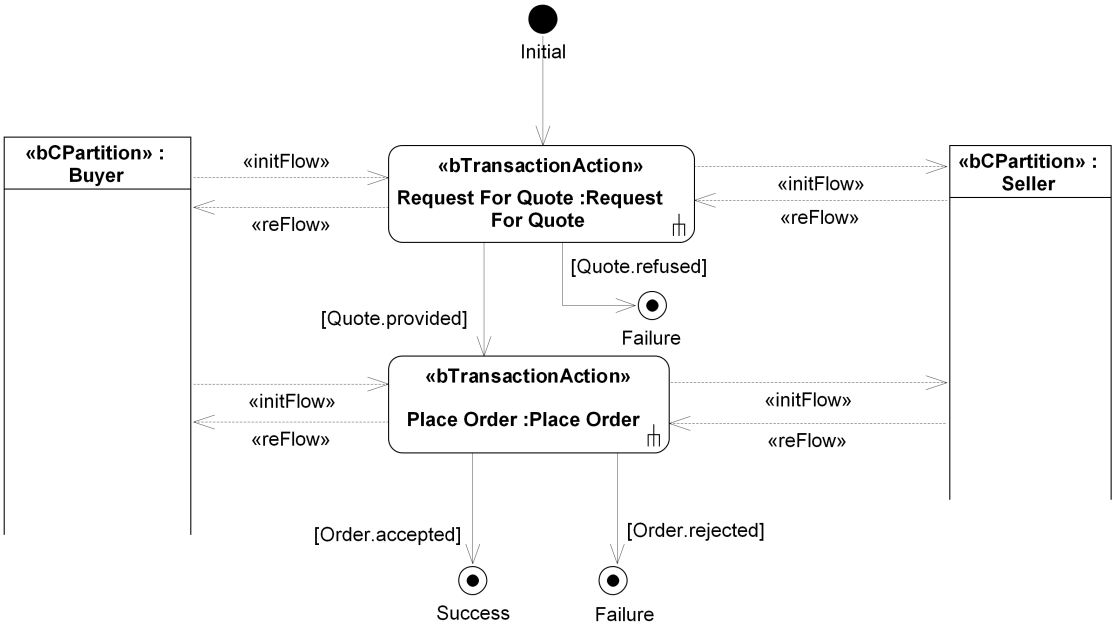

Figure 5.29: Order From Quote - Business Collaboration Protocol

Figure 5.29 shows an activity diagram of a *business collaboration protocol* `Order From Quote`. The *authorized roles* `Buyer` and `Seller` participating in the business collaboration are classified to their respective *business collaboration partitions*. The *authorized role* `Buyer` triggers the *business transaction call* `Request For Quote: Request For Quote`.

This initiating flow of information is mapped from the *authorized role* of the business partition `Buyer` to the *authorized role* `Quote Requestor` of the

business transaction call `Request For Quote`. The initiating flow of information is also mapped from the *authorized role* `Quote Requestor` of the *business transaction call* `Request For Quote` to the *authorized role* of the business partition `Seller`.

Since, the *business transaction* `Request for Quote` is a two-way *business transaction* pattern there exists a *responding information flow* between the *authorized roles* `Buyer` and `Seller`. The `Seller` responds to the *business transaction call* through a *responding information flow*, mapping to the responding *authorized role* `Quote Responder` of the *business transaction* `Request For Quote`. Similarly, a mapping through a *responding information flow* from the *authorized role* `Quote Responder` of the *business transaction call* `Request For Quote` to the *authorized role* of the *business collaboration partition* `Buyer` completes the a *business transaction call* of the *business collaboration*.

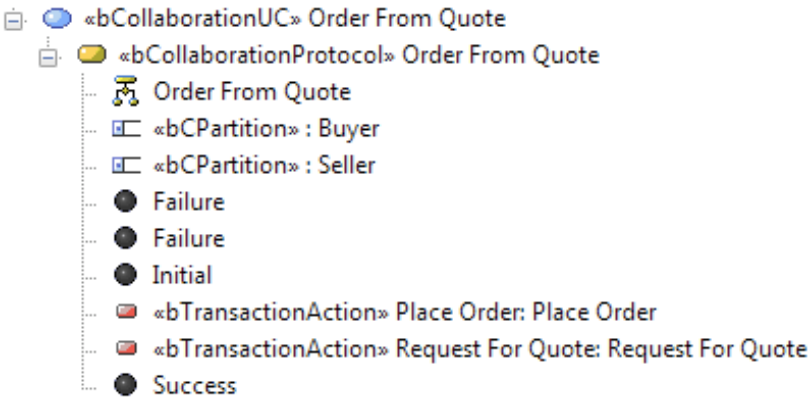

Figure 5.30: Business Collaboration Protocol - Tree View

If during the execution for the *business transaction call*, the quote was refused then the *business collaboration* is terminated, synchronizing the *business entity state* of *business entity* `Quote`. On the other hand, if the `Quote` was accepted then the next *business transaction call* is triggered i.e. `Place Order`. This *transaction call* is executed in the same manner as the previous transaction call, thereby manipulating the *business entity state* of the *business entity* `Order` based on if the order was accepted or rejected. With this the *business collaboration protocol* is terminated.

Figure 5.30 shows the tree view representation of *business collaboration use case* `Order From Quote`. All the artifacts such as *business collaboration partitions*: `Buyer`, `Seller` and *business transaction calls*: `Place Order`,

`Request For Quote` are created beneath the *business collaboration protocol*: `Order From Quote`.

In the *Appendix-Worksheets*, Table A.9 shows the *business collaboration protocol* worksheet template and Table 5.11 represents the worksheet for the *business collaboration protocol* `Order From Quote`.

Business Collaboration Protocol	
General	
Name	Order From Quote
Description	This business collaboration provides a quote requestor, the possibilities to request a quote for the required items and to provide the quote responder with the different possibilities to provide a formal quote or reject the quote. In case the quote is provided, the buyer is provided with the means to send the seller a list of items they wish to purchase and the seller is provided with the means to send the buyer a formal acceptance or rejection of the order placed.
Participants	
Participating Role	Quote Requestor, Quote Responder, Buyer, Seller
Included Business Transaction Actions	
Business Transaction Action	
Name	Request For Quote
Preceding Action(s) including transition condition	Register the customer
Initiating Role	Buyer
Reacting Role	Quote Requestor
Business Transaction Action	
Name	Place Order
Preceding Action(s) including transition condition	The requested quote is provided.
Initiating Role	Buyer (from the business collaboration protocol)
Reacting Role	Buyer (from the business transaction)

Table 5.11: Business Collaboration Protocol - Order From Quote

5.5.2.3 Tagged Values of stereotypes within the *bCollaborationV*

The stereotype *bCollaborationUC* generalizes from the stereotype *bProcessUC* of the *bDomainView* thereby inheriting all its tagged values.

- *definition*: Describes the purpose of a *business collaboration use case*.
- *preCondition*: A set of conditions that need to be fulfilled before the execution of the business collaboration.

- *postCondition*: A set of conditions that need to be fulfilled after the execution of the business collaboration.
- *beginsWhen*: Specifies specific business events that trigger the initiation of the business collaboration. It could also be a case where a certain state needs to be specified in order to trigger the business collaboration.
- *endWhen*: Specifies a list of business events or conditions that lead to the termination of the business collaboration.
- *actions*: Describes the activities that are performed during the execution of the business collaboration.
- *exceptions*: Lists all exception conditions that that causes the business collaboration to terminate before its normal completion.

Additionally, the *bCollaborationUC* also defines a tagged value *context* that describes the context (e.g., product, industry, official constraints etc.) of a business collaboration.

The stereotype *participates* contains a tagged value *interest* that describes a certain amount of interest, a *business partner* has in the business collaboration.

Furthermore, the stereotypes *bTransactionCall, bCollaborationCall, bNestedCollaboration* contain the tagged value *kind* which is an action with a call behaviour action kind.

The stereotype *bTransactionCall* contains the following tagged values:

- *isConcurrent:* If a *business transaction call* is set as concurrent, then more than one *business transaction call* of the same underlying *business transaction* can be open at one time for executing the same business collaboration with the same business partner type. If a *business transaction call* is not set as concurrent then only one *business transaction call* of the same underlying *business transaction* can be open at one time.
- *TimeToPerform:* The maximum duration of time required by the requesting *authorized role* to receive a reply from the initiated *business transaction call*. In other words, to receive a responding information envelope or an acknowledgement of processing if any, or an acknowledgement of processing if any, to the requesting business information envelope sent. The initiating *authorized role* needs to terminate the execution of the business activity and send a failure notification to the responding *authorized role* if the *business transaction call*, if not executed within a specific duration of time.

5.5.2.4 Summary of artifacts created within the *bCollaborationV*

Results of the artifacts created within the *business collaboration view* include the following:

- ***authorized roles*** : Actors participating in a business collaboration. It is used as a more generic term than *business partners* participating in *business processes*.

- ***business collaboration use case*** : A use case that extends the *business process use case* and captures the requirements of a business collaboration between two or more *authorized roles* along with its included *business transactions/business collaborations*.

- ***participates***: An association between an *authorized role* and a *business collaboration use case*. This association indicates that an input or output is provided by the *authorized role* to its associated *business collaboration use case*.

- ***business collaboration protocol***: An UML activity that describes the collaborative business process between two or more *authorized roles*. The choreography of *business processes* is described through *business transaction calls* and/or *business collaboration calls*. A flow of UML actions describe the business collaboration based on the requirements gathered from the *business collaboration use case*.

- ***business collaboration partition***: An UML partition in the *business collaboration protocol* that is classified to an *authorized role*. The *business collaboration partition* is left empty but in case of a two-way *business transaction* the *business collaboration partition* may contain a *nested business collaboration* which responds to a *business transaction call* which initiates a *nested business collaboration*.

- ***business transaction call***: An action that executes a called *business transaction*. It is defined within a *business collaboration protocol*. If the tagged value of the *business collaboration protocol* "isConcurrent" is set as true then the *business transaction call* can be executed more than once. Each *business transaction* must be called at least once by the *business transaction call*.

- ***business collaboration call***: An action that calls and executes a called *business collaboration protocol*. It is defined within a *business collaboration protocol*. Not every *business collaboration protocol* needs to be called by the *business collaboration call* however a *business collaboration protocol* may be called by multiple business collaboration calls.

- **nested business collaboration**: A business collaboration which is executed exactly once within a *business collaboration protocol*. It is defined within an *authorized role* of a *business collaboration protocol*. A *nested business collaboration* is executed after the responding *authorized role* of a *business transaction* receives a requesting information envelope, which is represented by an initiating flow and before responding to the initiating *authorized role* of the *business transaction call*, which is represented by a responding flow.

- ***initiating flow (initFlow)*** The initiating flow is an information flow that initiates the execution of the *business transaction* call as well as the execution at the responders end. In case of a *business transaction call*, the *business collaboration partition* is the source of the initiating flow and the *business transaction call* is the target. In the latter case, the *business transaction call* is the source of the initiation flow and the *business collaboration partition* responding to the *business transaction call* is the target. In both cases, a mapping is provided between the *business collaboration partition* initiating the *business transaction call* and the initiating role of the *business transaction* called by the *business transaction call*. At the same time a mapping is provided between the responding role of the *business transaction call* and the responding role of the *business collaboration protocol*. In case of a *nested business collaboration*, the target of the initiating flow is the *nested business collaboration* residing within the *business collaboration partition* and not the *business collaboration partition* itself.

- ***responding flow (reFlow)***: The *responding information flow* that completes a two-way *business transaction* at the *business transactions* calls end as well as the execution at the initiators end, that initiated the *business transaction call*. In case of the *business transaction call*, the source of the responding flow is the *business collaboration partition* and the target is the *business transaction call*. In special cases the source of the *responding information flow* is the *nested business collaboration* residing within the *business collaboration partition*. In the latter case, the source of the initiating flow is the *business transaction call* and the target is the *business collaboration partition*, initiating the *business transaction call*.

5.5.3 Business Realization View

5.5.3.1 Overview and purpose

UMM does not allow a direct association of *business partners* to its *business collaboration*. *Business realization view* is used to map *business partners* to *business collaborations*. The *business realization view* defines which *business partners* play which roles in which *business collaboration*. This allows different sets of *business partners* perform the same *business collaboration*. Through this, the *business realization view* is highly scalable, since it allows the same *business collaboration* to be performed between unlimited set of *business partners*. This decreases the complexity thereby increases reusability.

5.5.3.2 Step-by-Step modeling Guide

Step 1. ***Identify a business realization use case and map to its corresponding business collaboration use case.***

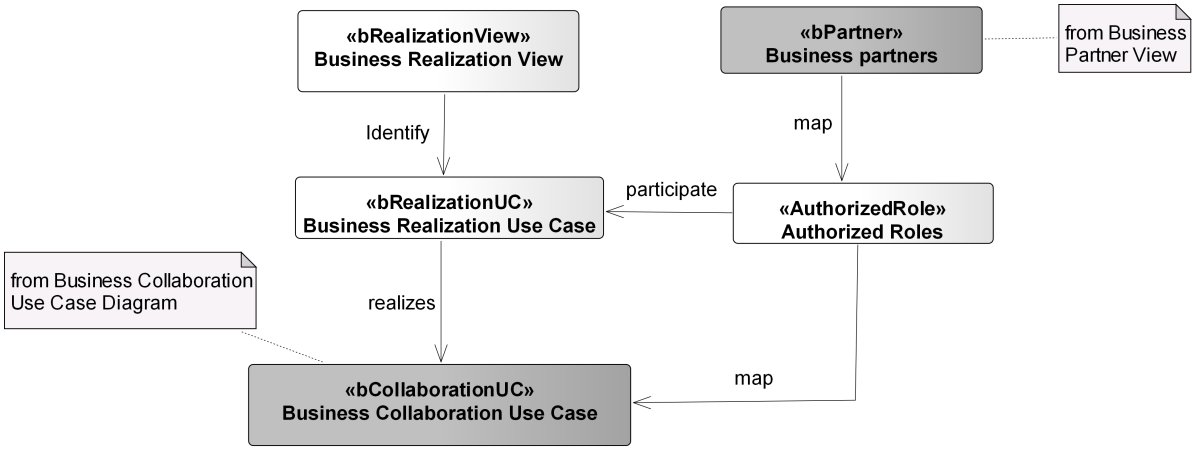

Figure 5.31: Workflow - Business Realization View

Overview and purpose
A *business realization use case* realizes exactly one *business collaboration use case* between a set of existing *business partners* from the *business partner view*. The *business realization use case* is an exact replica of its corresponding *business collaboration use case* modeling elements, except the fact that is it extended via mapping to its respective *business partners* and *authorized roles*.

Map business realization use case to business collaboration use case.
A *business realization use case* stereotyped as *BusinessRealizationUseCase (bRealizationUC)* is created beneath the *business realization view* and described with the help of a UML use case diagram. A *business realization view* should contain exactly one *business realization use case*. It is suggested to use the name of the *business collaboration use case* it realizes. A *business collaboration use case* can be realized through multiple *business realization use cases*. Every *business collaboration use case* need not be realized by a *business realization use case*.

Modeling elements of the *business collaboration use case* diagram which the *business realization use case* realizes are dragged in the use case diagram. Model elements include the *business collaboration use case*, *authorized roles* and the *participates* association that exists between the *business collaboration use case* and the *authorized roles*.

Authorized roles defined for the *business collaboration use case* are also created for the *business realization use case* and are linked through an association

stereotyped as *participates* to the *business realization use case*. Since two or more *authorized roles* participate in the *business collaboration use case*, the *business realization view* can also be composed of two or more *authorized roles*. It is recommended that the *authorized roles* participating in the *business realization use case* have the same name as that of the *authorized roles* participating in the *business collaboration use case*. The number of *authorized roles* in the *business realization use case* must be exactly the same number as that of its corresponding *business collaboration use case*. Each of the *authorized roles* created, are mapped to the *authorized roles* of the *business collaboration use case* through a dependency stereotyped as *mapsTo*.

In order to indicate that a *business realization use case* realizes a certain *business collaboration use case*, we link the *business realization use case* to the *business collaboration use case* with a realization dependency stereotyped as *realize*.

Step 2. ***Map business partners to authorized roles.***

Overview and purpose
Business partners defined in the *business requirements view* that embark a certain role within the *business realization* are identified here through the *authorized roles* who take part in the *business collaboration use case*. For each *authorized role* of the *business realization use case*, its relevant *business partners* are identified.

Model the mapping
Business partners are mapped to their corresponding *authorized role* of the *business realization use case* via a dependency stereotyped as *mapsTo*. A *business partners* may play different roles in different *business collaboration use case* but it takes over, exactly one role within a *business realization use case*.

Figure 5.32 shows the *business realization use case* `Order From Quote` that realizes the *business collaboration use case* `Order From Quote`. All the artifacts from the *business collaboration use case* diagram `Order from Quote` are dragged into the *business realization use case* `Order From Quote`. This includes the *authorized roles* `Buyer` and `Seller` along with the stereotype *business collaboration use case* `OrderFromQuote` and *participates* association between them.

A dependency stereotyped as *realizes* is linked from the *business realization use case* `Order From Quote` to the *business collaboration use case*. Two more *authorized roles* with the same name as in the *business collaboration use case*, `Buyer` and `Seller` are defined in the *business realization view*. The `Buyer` of the *business realization use case* is mapped to the `Buyer` of the *business collaboration use case* via

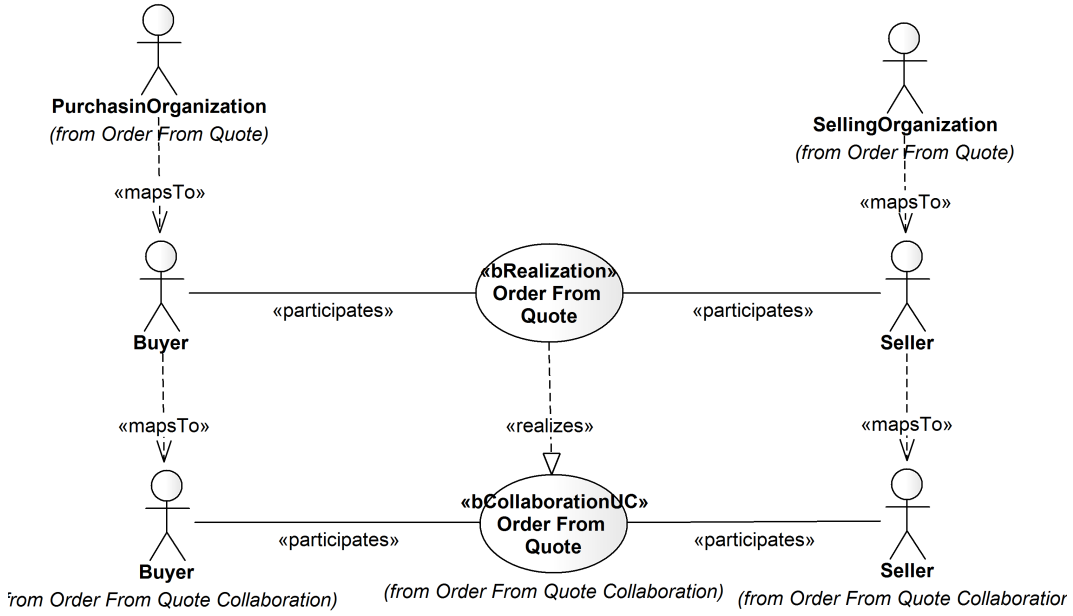

Figure 5.32: Order from Quote - Business Realization Use Case

mapsTo association. The same applies to the *authorized role* `Seller`. The `Buyer` and the `Seller` are associated to the *business realization use case* `Order From Quote` via `participates` association.

Further on, two *business partners*, `Purchasing Organization` and `Selling Organization` take up certain roles, as `Buyer` and `Seller` in the *business realization use case*. Hence, these *business partners* are mapped to their respective *authorized roles* via *mapsTo* dependency. The `Purchasing Organization` is mapped to the `Buyer` via *mapsTo* dependency and the `Selling Organization` is mapped to the `Seller` via *mapsTo* dependency.

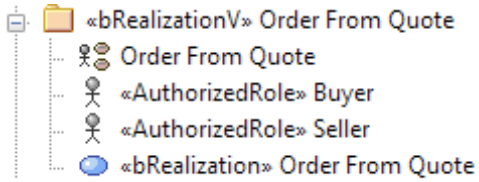

Figure 5.33: Business Realization View - Tree View

Figure 5.33 shows the same artifacts created in the *business realization use case* diagram, in the form of a tree view. This includes two *authorized roles*- `Buyer` and `Seller` participating in the *business realization use case* and the *business realization use case* `Order From Quote`.

5.5.3.3 Tagged Values of stereotypes within the *bRealizationView*

The *business realization use case* does not contain any tag definitions rather are defined in its corresponding *business collaboration use case*.

5.5.3.4 Summary of artifacts created in *bRealizationView*

Results of the artifacts created within the *business realization view* include the following:

- ***business realization use case***: It is a use case diagram that realizes a *business collaboration* use case for a specific set of *business partners*.
- ***authorized roles***: Actors whose terms are more generic than *business partners*. In a *business realization use case* they are mapped to the *authorized roles* of the *business collaboration* use case.
- ***maps to***: A dependency that represents two cases. The first one is where *business partners* plays a certain specific role in the *business realization use case*. Secondly, the *authorized role* of a *business collaboration use case* takes a specific role in the *business transaction use case* or *business collaboration use case* which is called.

5.5.0.4 Summary of artifacts created within the *bChoreographyV*

The *bChoreographyView* results in the following artifacts:

- The ***business transaction view*** is a package that defines the order of business information flow between exactly two *authorized roles* that leads to a synchronized state of *business entities* at both sides of the *business transaction*.
- The ***business collaboration view*** is a package that models a choreography based on multiple *business transactions*. It is a container of artifacts that defines the choreography between multiple participants.
- The ***business realization view*** is a package that realizes a business choreography between different sets of *business partners* from a business collaboration.

5.6 Business Information View

5.6.0.1 Overview and purpose

The *business information view* contains all the artifacts required to define the structure of a business document or in other words describes the *business information envelopes*, exchanged between *authorized roles* in *business transactions*. Each of information envelopes exchanged in *business transactions* leads to a *business information view* describing the envelopes document structure. The current UMM modeling approach does not mandate the user to a specific business information modeling approach rather strongly suggests the use of UN/CEFACT's UML profile for Core Components to model business messages.

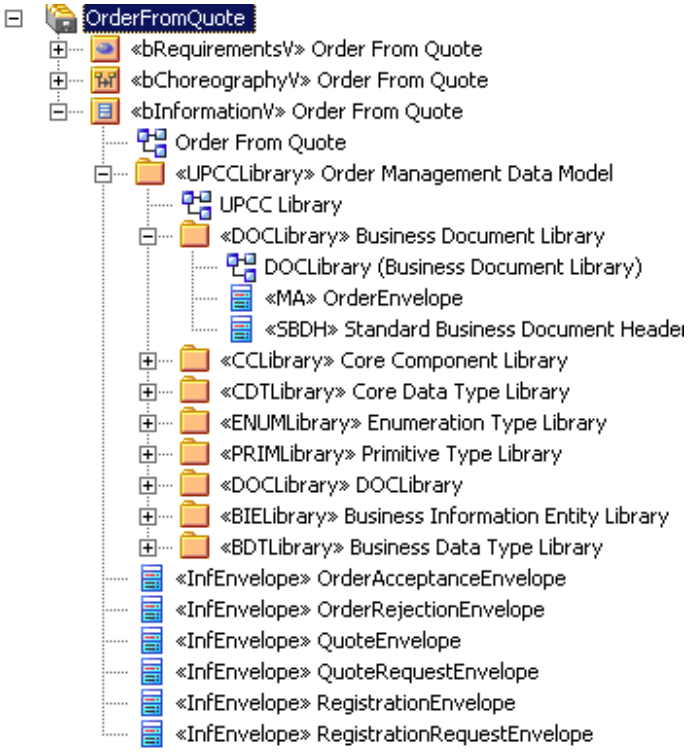

Figure 5.34: Business Information View - Tree View

5.6.0.2 Step-by-Step modeling guide

Step 1. ***Create a business information envelope***

A business information envelope stereotyped as *InformationEnvelope (InfEnvelope)* from the UMM base module is created beneath the *business information view*. The *InformationEnvelope* is a subtype of an abstract class named business information stereotyped as *bInformation (BusinessInformation)*. *BInformation* is used to realize abstract business information exchanged between participants in a *business transaction*. The *business information view* must contain at least one *InformationEnvelope* any may contain any other business document modeling artifacts. The *message assembly* from the UPCC model is associated to the *InformationEnvelope* using a standard UML aggregation.

Figure 5.34 shows a tree view representation of the *business information view* `Order from Quote` containing a *UPCCLibrary* `Order Management Data Model` and a list of *Information Envelopes* representing concrete business messages.

5.6.0.3 Summary of artifacts created within the *bInfView*

A result of the artifacts created within the *business Information model* are:

- The ***business information*** is an abstract class that realizes the business message exchanged between *authorized roles* during a *business transaction*.
- The ***information envelope*** is a class and a subtype of *business information* that represents the exact information exchanged during a *business transaction*. Associations are used to connect *business information envelopes*.

5.6.0.4 Summary of artifacts created within the *bCollModel*

A result of the artifacts created within the *business collaboration model* are:

- the ***business requirements view*** is a package that gathers all the requirements needed during the *business collaboration* modeling process,
- the ***business choreography view*** is a package that designs the collaboration between multiple participants.
- the ***business information view*** is a package that models the business documents exchanged between *business partners* during the *business choreography* process.

CHAPTER 6

UPCC User Guide

6.1 Overview of UN/CEFACT's Core Components

In the past, business documents were exchanged between applications, based on static message definitions due to a lack of interoperability between *business partners*.

CCTS distinguishes between two packages that reflect the basic concepts of the core and business context: *Core (Core Components)* and *Business (Business Information Entities)*.The basic concept in the CCTS specification is a *Core Component*. *Core components* are the main pillars independent of any specific business domain, used as standard building blocks for business documents. They represent reusable parts with a common semantic meaning and are used to define business documents. They do not represent any specific business scenario and hence can be used for any given purpose. When *core components* are adapted to a certain business environment, they are called as *Business Information Entities(BIE)*. The *BIE's* are derived from the *Core Components* by restrictions and a used for a specific business domain.

The relationship between *core components* and *BIE's* is shown through a conceptual model of the CCTS, in Figure 6.1. The right hand side of the diagram shows that the *core component* distinguishes between *Aggregate Core Component (ACC), Association Core Component (ASCC)* and *Basic Core Component (BCC)*. An *ACC* is an entity that consists of several *BCC's*. In order to establish relationships between *ACC's* the concept of so called *ASCC's* are used. The data types of the *Basic Core Component* are referred as the *Core Data types (CDT)*.

If *core components* are used for a specific business context, *Business Information Entities (BIE's)* are defined as shown in the right hand side of Figure 6.1. Similar to *core components*, the *BIE's* are distinguished between *Aggregate Business Information Entity (ABIE), Association Business Information Entity (ASBIE)* and *Basic Business*

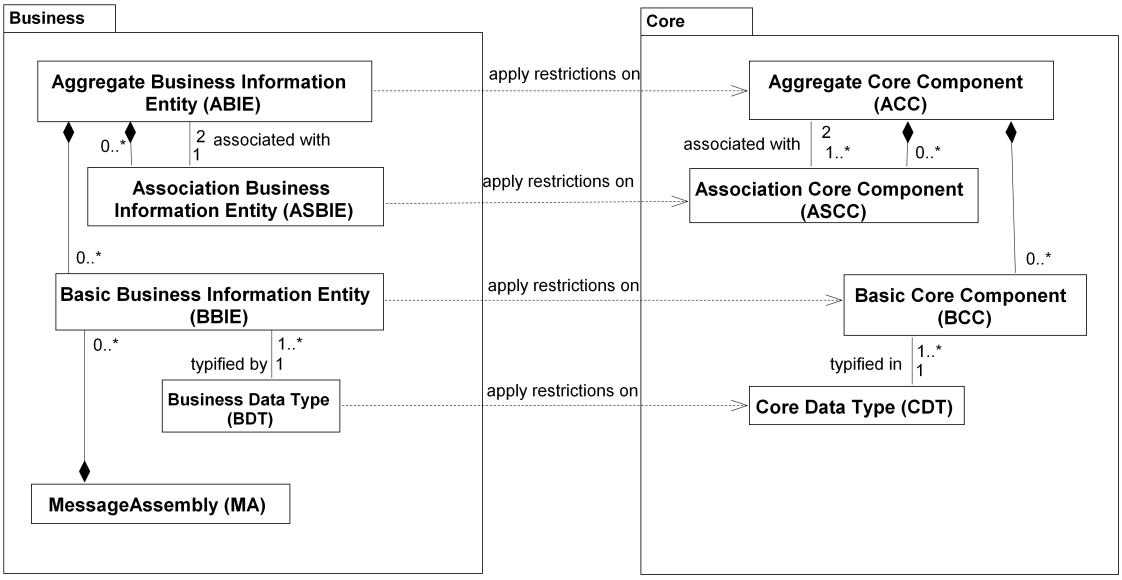

Figure 6.1: Relationship between Core Components and Business Information Entities

Information Entity (BBIE). The data types of *Basic Business Information Entities* are referred as the *Business Data Types (BDT's)*.

Table 6.1 gives an overview of the CCTS concepts.

Property type	**Core Component**	**Business Information Entity**
Object Type	Aggregate Core Component (ACC)	Aggregate Business Information Entity(ABIE)
Simple Property	Basic Core Component (BCC)	Basic Business Information Entity(BBIE)
Complex Property	Association Core Component (ASCC)	Association Business Information Entity(ASBIE)
Simple Property Data type	Core Data Type (CDT)	Business Data Type (BDT)

Table 6.1: Properties of Core Components

6.2 UPCC Library

6.2.0.5 Overview and purpose

UPCC library is used as a container of packages serving different purposes for creating a UPCC model. UPCC library forms the root element of the UPCC model.

6.2.0.6 Step-by-Step modeling guide

UPCC library is a package that extends the *business library* from the UMM base module and is stereotyped as *UPCCLibrary*. It is created and integrated within the *business information view* of the UMM model where all business information relevant artifacts are summarized. Based on the modelers requirements for UMM, the following libraries are created within the UPCC library for which every artifact type a dedicated library is assigned.

1. *Core Component Library*: A package stereotyped as *Core Component Library (CCLibrary)* is used as a container of objects named as core components. They are used as building blocks for business document modeling independent of any business context. The *UPCCLibrary* can contain any number of *CCLibrary's* or no *CCLibrary's* at all.
2. *Core Data Type Library*: A package stereotyped as *Core Data Type Library (CDTLibrary)* is used as a container of data types that represent the core component properties independent of business context. The *UPCCLibrary* can contain any number of *CDTLibrary's* or no *CDTLibrary's* at all.
3. *Primitive Library*: A package stereotyped as *Primitive Library (PRIMLibrary)* is used to contain a list of primitive types, defined by the Core Component Data Type Catalogue 3.0 [41]. The *UPCCLibrary* can contain any number of *PRIMLibrary's* or no *PRIMLibrary's* at all.
4. *Enumeration Library*: A package stereotyped as *Enumeration Library (ENUMLibrary)* is used to contain a set of enumeration values which are used to depict code list and a set of Identifier Schemes types that are used to depict non-enumerated identifier scheme types. The *UPCCLibrary* can contain any number of *ENUMLibrary's* or no *ENUMLibrary's* at all.
5. *Business Information Entity Library*: A package stereotyped as *Business Information Entity Library (BIELibrary)* is used as a container of core components related to a specific business context. The *UPCCLibrary* can contain any number of *BIELibrary's* or no *BIELibrary's* at all.
6. *Business Data Type Library*: A package stereotyped as *Business Data Type Library (BDTLibrary)* is used as a container of core data types related to a certain

business context. The *UPCCLibrary* can contain any number of *BDTLibrary's* or no *BDTLibrary's* at all.

7. *Business Document Library*: A package stereotyped as *Business Document Library (DOCLibrary)* is used as a container of exactly one business document. A *DOCLibrary* exists exclusively only for one business document. The *UPCCLibrary* can contain any number of *DOCLibrary's* or no *DOCLibrary's* at all.

Furthermore, a class is used to define constraints on specific artifacts.

UsageRule: A class stereotyped as *UsageRule* defines constraints describing the conditions that apply on the following artifacts from their respective libraries: *aggregate core component, basic core component, association core component, aggregate business information entity, association business information entity, basic business information entity, core data type, business data type, content component, supplementary component, enumeration type, identifier scheme type* and *primitive types*.

Additionally, two dependencies are used in the libraries of UPCC:

isEquivalent: Artifacts such as *Aggregate Core Component*, *Aggregate Business Information Entity*, *Core Data Type*, *Business Data Type*, *Primitive Type*, *Enumeration Type* and *Identifier Scheme Type* from their respective libraries can be represented in different languages. In order to denote that a given artifact is equivalent to another artifact excluding the language part, a dependency stereotyped as *isEquivalent* is used.

basedOn: In order to indicate the "derivation-by-restriction" mechanism of core components artifacts, a dependency stereotyped as *basedOn* is used for which the following relationships are encouraged:

- between an *aggregate business information entity* of the *BIELibrary* and its corresponding *aggregate core component* of the *CCLibrary*.
- between a *business data type* of the *BDTLibrary* and its corresponding *core data type* of the *CDTLibrary*.
- between an *enumeration* representing a business code list and an *enumeration* representing a core code list.
- between an *identifier scheme* representing a business identifier scheme and an *identifier scheme* representing a core identifier scheme.

Together with the core components, the UN/CEFACT defines Naming and Design Roles (NDR) [40], allow a unique mapping of core component based business document models to XML schema representation. The NDR are guidelines for generating

XML schema definitions(XSD) out of a conceptual business documents. With the core components defined at the conceptual level, every release does not require a complete re-engineering standard. Just the NDR's are updated. These NDR's are mapped only from the *BIE's* that represent a unique representation of core components in XML. The generated XSD schema is then used to validate the actual information exchanged in the *business process*.

A set of stereotypes such as *primitives*, *enumeration types* and *identifier scheme types* extend from the supertype abstract class *value domain*. The *value domain* contains the following tagged values:

- *definition:* Free text form describing the *PRIM/ ENUM/ IDSCHEME* in more detail.
- *dictionaryEntryname:* A unique official dictionary entry name of the *PRIM/ ENUM/ IDSCHEME* based on the dictionary entry name rules of the CCTS [31].
- *uniqueIdentifier:* This identifier uniquely represents the *PRIM/ ENUM/ IDSCHEME* in the registry. The ITU-T Rec. X.667|ISO/IEC9834-8 Universally Unique Identifier (UUID) scheme [52] is used to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the library to which it belongs.
- *versionIdentifier:* A unique version identifier that represents the version of the *PRIM/ ENUM/ IDSCHEME* and which evolves over time of different instanzes.
- *minorVersionIdentifier:* This identifier represents a series of minor changes within the major version of the *PRIM/ ENUM/ IDSCHEME* represented by the *versionIdentifier*.
- *revisionIdentifier:* This identifier represents the revision version in addition to the *versionIdentifier* and *minorVersionIdentifier* of the *PRIM/ ENUM/ IDSCHEME*.
- *usageRule:* A constraint represented in a free text format describing the conditions that apply to *PRIM/ ENUM/ IDSCHEME*. In order to avoid inconsistency between the usage rule and OCL constraint, a unique identifier should be used for both.

6.2.0.7 Tagged values created within the UPCC Library

The stereotypes *UPCCLibrary, CCLibrary, CDTLibrary, BIELibrary, BDTLibrary, PRIMLibrary* and *ENUMLibrary* inherit all the tagged values from the *business library*. Additionally, the UPCC library contains the following tagged values:

- *namespacePrefix:* A prefix applied for namespaces of deployment artifacts resulted due to the sub packages created within the UPCC library.

- *minorVersionIdentifier:* An identifier representing a series of minor changes within the major version of the *UPCC library* represented by the *versionIdentifier:*.
- *revisionIdentifier:* An identifier representing the revision version in addition to the *minorVersionIdentifier* of the *UPCC library*.
- *baseURN:* A valid Uniform Resource Name (URN) that represents the namespace of a *blibrary* along with the local name (name of the instance) and the version of the instance.

Furthermore, the stereotype *UsageRule* contains the following tagged values:

- *conditionType:* A condition which can be one of the following types: pre-condition, post-condition.
- *constraint:* An OCL constraint describing the condition applied on the *UsageRule*.
- *constraintType:* The type of constraint defined. Since the UPCC is built on UML, OCL is the only type of constraint used here.
- *uniqueIdentifier:* An identifier that is unique while being a part among all usage rules for the library.

6.2.1 Core Component Library

6.2.1.1 Overview and purpose

The *core component library* is a UML representation of the UN/CEFACT Core Component Library which is used as a container of core components. Core components are semantic building blocks that create a correct and relevant business information exchange package. They are the cornerstone for creating interoperable business process models and business documents. They are syntax and platform independent which are standardized through regular spread sheets and are used to create context specific *Business Information Entities*; *refer Section 6.2.5*.

6.2.1.2 Step-by-Step modeling guide

A business document modeler must not create new core components, rather reuse the existing core components from the UN/CEFACT Library. If user-specific core components are created then the consistency to the UN/CEFACT library is lost. Business document modelers may search and retrieve core components which are needed from the UN/CEFACT library. The *CCLibrary* distinguishes between three kinds of core components:

- **Aggregate core component**: A class stereotyped as *Aggregate Core Component (ACC)* gives a distinct meaning to closely related business informations. Any number of *ACC's* can be created within the *CCLibrary* or no *ACC's* at all. The name of the *ACC* is same as that of the object class term as defined in the CCTS.

- **Basic core component**: A property representing a business characteristic of an *ACC*. It is stereotyped as *Basic Core Component (BCC)* and created beneath its respective *ACC*. The name of the *BCC* should be the same as that of the basic core component property term excluding the object class and representation terms as defined in the CCTS. In order to ensure uniqueness, if an *ACC* has two *BCC's* with the same property term then the representation term should be added to the UML attribute name to ensure uniqueness. An *ACC* can contain zero to multiple *BCC's*. The *BCC* is typified using the *Core Data Type* from the *Core Data Library* (refer section 6.2.2).

- **Association core component**: An UML association of AggregationKind= shared, that associates exactly two *ACC's*. It is stereotyped as *Association Core Component (ASCC)* and created beneath its respective *ACC* . The associated *ACC* represents a complex property of the associating *ACC*. The role name of the associated *ACC* uniquely represents the *ASCC* and is the same at that of the associating core component property term defined in the CCTS. Any number of *ASCC's* can be created within the *ACC* or no *ASCC's* at all.

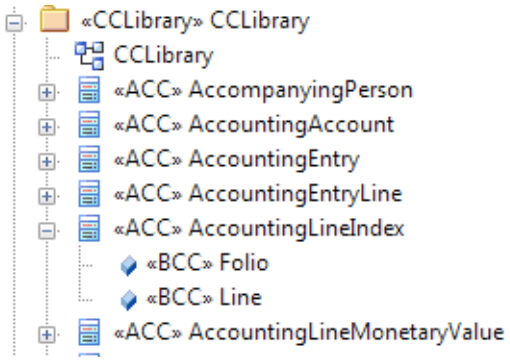

Figure 6.2: Core Component Library - Tree View

In Figure 6.2, the stereotype `CCLibrary` contains an *ACC* `AccountingLineIndex` and two *BCC's*: `Folio` and `Line`.

6.2.1.3 Tagged values of stereotypes within the CC Library

The stereotypes *ACC*, *BCC* and *ASCC* inherit from the abstract stereotype *Core Component* thereby inheriting the following tagged values:

- *businessTerm:* A synonym used in business through which the *ASCC/ACC/BCC* are commonly known. An *ASCC/BCC/ACC* may use different business terms as well.
- *languageCode:* The code that states which language needs to be used. The Internet Engineering Task Force (IETF) RFC 3066 of January 2001 [51] is used as the source for the code values.
- *uniqueIdentifier:* An identifier that uniquely represents the *ASCC/BCC/ACC* in the registry. The ITU-T Rec. X.667|ISO/IEC9834-8 Universally Unique Identifier (UUID) [52] scheme is used to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the library to which it belongs.
- *versionIdentifier:* A unique version identifier that represents the version of the *ASCC/BCC/ACC* and evolves over time over different instances.
- *definition:* Free text form describing the *ASCC/BCC/ACC* in more detail.
- *dictionaryEntryname:* A unique official dictionary entry name of the *ASCC/BCC/ACC* based on the dictionary entry name rules of the CCTS [31].
- *usageRule:* A constraint representing a free text format describing the conditions that apply to an *ASCC/BCC/ACC*. In order to avoid inconsistency between the *usage rule* and OCL constraint, a unique identifier should be used for both of them.

Additionally, the *BCC* and *ASCC* contain the following tagged values:

- *sequencingKey:* Unique sequencing algorithms may be used by software and storage applications to change the standard defined order of a *BCC* or *ASCC* within an *ACC* respectively. A unique sequencing key is entitled to each *BCC* or *ASCC* within an *ACC* to ensure that a desired order is preserved respectively.

6.2.1.4 Summary of artifacts created within the CC Library

The *CCLibrary* is a container of core components. Results of artifacts in the *CCLibrary* include:

- ***Aggregated Core Component***: A class containing business related information carrying a distinct meaning.

- ***Basic Core Component***: A property describing the business characteristics of an *ACC*.
- ***Association Core Component***: An association associating exactly two *ACC's*. The relations between the *ACC's* are specified through *ASCC's*.

6.2.2 Core Data Type Library

6.2.2.1 Overview and purpose

The *core data type library* is used as a container of complex data type elements called *core data types*. In a *core data type* the main content as well as additional information of a *core component* is stored.

6.2.2.2 Step-by-Step modeling guide

A business document modeler must not create new *core data types*, rather reuse the existing ones from the UN/CEFACT Data Type Catalogue [41]. If user-specific *core data types* are created then the consistency to the UN/CEFACT's *core components* is lost.

A *core data type* is a UML class stereotyped as *Core Data Type (CDT)* is created beneath the *CDTLibrary*. It is a complex data type element where the actual content of information of a *core component* is stored in the content component, a property stereotyped as *Content Component (CON)*. Additional meaning about the *CON* is stored in the supplementary component, a property stereotyped as *Supplementary Component (SUP)*. A *CDT* should already be approved and defined in the Core Component Data Type Catalogue 3.0 [41]. A *CDT* class name is equivalent to the data type term defined in the CCTS. Exactly one *CON* and any number of *SUP's* can be defined within the *CDT*. The *CON* as well as the *SUP's* are typified using *primitive Types (PRIM)*, *enumeration types (ENUM)* or identifier scheme types (IDSCHEME) which are already defined in the UN/CEFACT Data Type Catalogue 3.0 [41].

Figure 6.3 shows a tree view representation of the *CDTLibrary*, containing several *CDT's*. Taking into consideration, the *CDT* `Amount` created beneath the `CDTLibrary` contains a *CON* `Content` and *SUP's* `Currency` and `CurrencyCodeListVersion`.

6.2.2.3 Tagged values of stereotypes within the CDT Library

The *CDT* as well as the *CON* and *SUP's* generalize from the abstract *core data type property* containing the following tagged values:

- *businessTerm:* A synonym used in business through which the *CDT/CON/SUP* are commonly known. A *CDT/CON/SUP* may use different business terms as well.

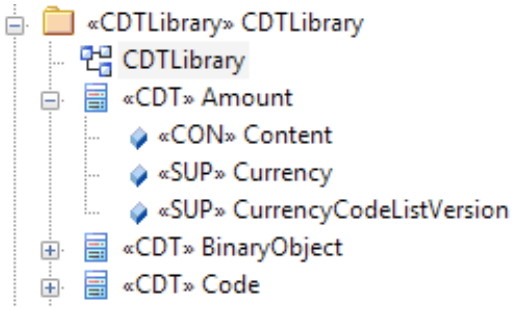

Figure 6.3: Core Data Library - Tree View

- *definition:* Free text description describing the *CDT/CON/SUP* in more detail.
- *dictionaryEntryname:* A unique official dictionary entry name of the *CDT/CON/SUP* based on the dictionary entry name rules of the CCTS [31].
- *languageCode:* The code that states which language needs to be used for the *CDT/CON/SUP*. The Internet Engineering Task Force (IETF) RFC 3066 of January 2001 [51] is used as the source for the code values.
- *uniqueIdentifier:* The identifier uniquely represents the *CDT/CON/SUP* in the registry. The ITU-T Rec. X.667|ISO/IEC9834-8 Universally Unique Identifier (UUID) scheme [52] is used to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the library to which it belongs.
- *versionIdentifier:* A unique version identifier that represents the version of the *CDT/CON/SUP* and evolves over time over different instances.
- *usageRule:* A constraint in a free text format describing the conditions that apply to the *CDT/CON/SUP*. In order to avoid inconsistency between the *usage rule* and OCL constraint, a unique identifier should be used for both.
- *modificationAllowedIndicator*: An indicator that applies restrictions on enumerations through *BDT's* in order to set the value domain of *CON/SUP*. If the modifications are not desired then the indicator is set as false else it is set as true.

6.2.2.4 Summary of artifacts created within the CDT Library

The *CDTLibrary* is a container of *CDT's*. Results of the *CDTLibrary* include the following:

- ***Core Data Type***: A class that represent a complex data type whose name is already approved and defined by the UN/CEFACT Data Type Catalogue 3.0 [41].

- ***Content Component***: A property that contains the actual content of the *CDT*. The *CON* is typified using one of the defined *primitive types, enumeration types* and *identifier scheme types* from the UN/CEFACT Data Type Catalogue 3.0 [41].
- ***Supplementary Component***: A property that contains additional meaning to the *CDT*. The *SUP* is typified using one of the defined *primitive types, enumeration types* and *identifier scheme types* from the UN/CEFACT Data Type Catalogue 3.0 [41].

6.2.3 Primitive Type Library

6.2.3.1 Overview and purpose

The *primitive type library* is used as a container of *primitive data types*. These primitive types are used to set the value domains of *CON's* and *SUP's* of *CDT's* and *BDT's*. A *primitive data type* has two specializations: Enumeration Types and Identifier Scheme Types (*refer section 6.2.4*). Primitive types may be restricted using the concept of facets which are specified in the CDT Data Type Catalogue 3.0 [41]. The facets per primitive type are specified using constraints.

6.2.3.2 Step-by-Step modeling guide

A primitive data type stereotyped as *Primitive Type (PRIM)* is created beneath the *PRIMLibrary*. The *primitive data types* are defined in the Core Component Data Type Catalogue 3.0 [41] and are `Binary`, `Boolean`, `Decimal`, `Double`, `Float`, `Integer`, `NormalizedString`, `String`, `TimeDuration`, `TimePoint` and `Token`. The *primitive library* can contain any number of *primitive data types* or no *primitive data types* at all.

The `Order Management Data Model` contains *primitive types* shown in Figure 6.2.

6.2.3.3 Tagged values for stereotypes within the PRIM Library

The *primitive data types* inherit the tagged values from the abstract type *Value Domain*. Additionally, the *PRIM* contains the following tagged values:

- *businessTerm:* A synonym used in business through which the *PRIM* is commonly known. A *PRIM* may use different business terms as well.
- *pattern:* A constraint represented as a regular expression on the value space for constraining the lexical space to literals that match a specific pattern.

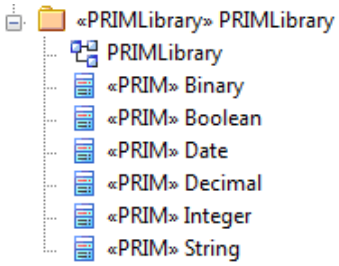

Figure 6.4: Primitive Library - Tree View

- *fractionalDigits:* Restrict the value space to those values that can be represented lexically using the most fractional digits after the decimal point.
- *length:* A non-negative integer representing a unit of length that a value of a given type should have .
- *maximumExclusive:* The exclusive upper bound value for an ordered data type of the value space.
- *maximumInclusive:* The inclusive upper bound value for an ordered data type of the value space.
- *maximumLength:* A non-negative integer that represents a maximum unit of length that a value of a given type should have.
- *minimumExclusive:* The exclusive lower bound value for an ordered data type of the value space.
- *mimimumInclusive:* The inclusive lower bound value for an ordered data type of the value space.
- *mimimumLength:* A non-negative integer that represents a minimum unit of length that a value of a given type should have.
- *totalDigits:* Restricts the value space to the lexically represented values using at most totaldigits. Additional zero digits or trailing fractional digits are also permitted.
- *whitespace:* Indicates how whitespace characters (line feeds, tabs, spaces, carriage returns) are handled when generating deployment artifacts out of the core component model.
- *enumeration:* Restricts the values of *primitive type* to a set of permitted values. This is in contrast to the ENUM type which represents a code list maintained by the code list agency.
- *languageCode:* The code that identifies which language needs to be used for the *CDT/CON/SUP*. The Internet Engineering Task Force (IETF) RFC 3066 of January 2001 [51] is used as the source for the code values.

6.2.3.4 Summary of artifacts created within the PRIM Library

The *PRIMLibrary* is a container of *primitive types*. Results of the *PRIMLibrary* include:

- ***Primitive Type:*** A datatype typified for the *CON's* and *SUP's* of the *CDT's* and *BDT's*.

6.2.4 Enumeration Type Library

6.2.4.1 Overview and purpose

The *ENUMLibrary* restricts the *PRIM's* to a set of enumerated values as well as define a specific template that a *PRIM* must follow. A *ENUMLibrary* is used as a container of enumeration data types. *ENUM* and *IDSCHEME* are used to set restrictions in the *CON's* or in the *SUP's* of the *CDT's* and *BDT's*.

6.2.4.2 Step-by-Step modeling guide

When the *primitive types* are confined to a specific set of values such as the Code List e.g. Phone Codes, the concept of enumeration type is used. The *Enumeration Type* stereotyped as *ENUM* is created beneath the *Enumeration Library*. With the concept of UML aggregations, a union of different enumerations can be made.

The *ENUM* entries stereotyped as *CodelistEntry* carry additional meta-information about the code list entry. *CCTS_ DT_3p0* is an *ENUM* that represents the allowed *primitive types*: `Binary`, `Boolean`, `Decimal`, `Double`, `Float`, `Integer`, `NormalizedString`, `String`, `TimeDuration`, `TimePoint` and `Token`, as defined in the UN/CEFACT Data Type Catalogue 3.0 [41].

When a *PRIM* restricts a value and follows a specific pattern e.g. a car plate number then the concept of *Identifier Scheme Types* are used. They are stereotyped as *IDSCHEME's* and are created beneath the *Enumeration Library*.

The *ENUMLibrary* can contain any number of *ENUM's* or *IDSCHEME's*. A tree view representation of a *ENUMLibrary* is shown in Figure 6.5.

6.2.4.3 Tagged values of stereotypes within the ENUMLibrary

The *ENUM* as well as the *IDSCHEME* inherits the tagged values from the abstract stereotype *Value Domain*. Additionally, the *ENUM* contains the following tagged values:

- *businessTerm:* A synonym used in business through which the *ENUM* is commonly known. A *ENUM* may use different business terms as well.
- *enumerationURI:* Identifies the physical representation of the an existing schema or a list artifact.

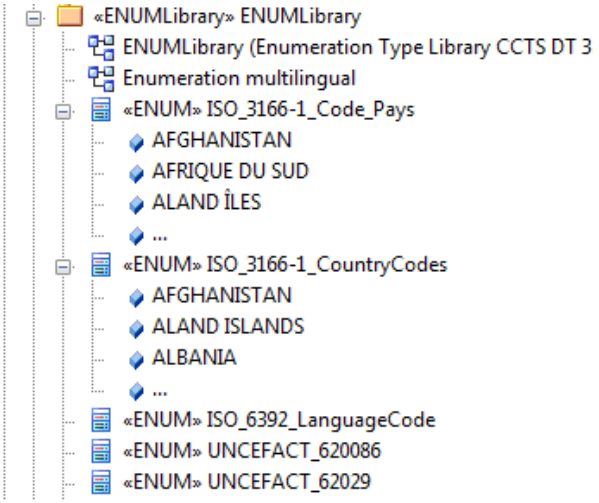

Figure 6.5: Enumeration Library - Tree View

- *languageCode:* The code that states which language needs to be used for the *ENUM* . The Internet Engineering Task Force (IETF) RFC 3066 of January 2001 [51] is used as the source for the code values.

Likewise, the *CodelistEntry* contains the following tagged values

- *codeName:* This field represents the name of a certain code e.g. “+43” in a phone codelist.
- *status:* This field is used to add meta information to a code lists entry e.g. “created”, “marked for deletion” etc.

Moreover, the *IDSCHEME* contains the following tagged values:

- *businessTerm:* A synonym used in business through which the *IDSCHEME* is commonly known. A *IDSCHEME* may use different business terms as well.
- *identifierSchemeAgencyIdentifier:* If an identifier scheme is maintained or owned by a specific organization, this tagged value serves as the unique identifier for the organization (e.g. UN/CEFACT). UN/CEFACT recommends using UN/CEFACT Agency Identifier code List (Data Element 3055) in the latest version of the UN/CEFACT directory.
- *IdentifierSchemeAgencyName:* The name of the organization maintaining the identifier scheme. e.g. United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT).

- *pattern:* A regular expression that defines the exact production rules for values according to the identifier scheme.
- *restrictedPrimitive:* This mandatory field indicates which primitive type the current identifier scheme type restricts. The value of this tagged value must be one of the approved primitive types from the the UN/CEFACT's Data Type Catalogue 3.0 [41] and defined in the stereotype *CCTS_ DT_3p0*.
- *modificationAllowedIndicator:* Indicates if restrictions are undesirable to be applied on *identifier schemes* by the *BDT's*.

6.2.4.4 Summary of artifacts created within the ENUMLibrary

The *ENUMLibrary* is a container of *ENUM's* and *identifier scheme types*. Results of the *ENUMLibrary* include

- ***Enumeration Type:*** A class that defines value domains of *CON* and *SUP's*. UN/CEFACT has defined a set of *PRIM's* which are already defined in the UN/CEFACT's Data Type Catalogue 3.0 [41].
- ***Code List Entry*** is an enumeration literal which is used to stereotype the entries of an enumeration having the stereotype *ENUM*.
- ***IDSCHEME:*** A datatype that restricts a *primitive data type* by representing a code list. *Enumeration types* are used to set restrictions on *CON's* or *SUP's* of *CDT's* and *BDT's*.
- ***CCTS_ DT_3p0:*** An enumeration that represents the allowed *primitive types* as defined in the UN/CEFACT Data Type Catalogue 3.0 [41].

6.2.5 Business Information Entity Library

6.2.5.1 Overview and purpose

In order to define a business document, the modeler takes the generic core components and tailors them to a specific business context and they are called Business Information Entities (BIE). Applying on a business context is a way of qualifying and selecting core components for a particular use in business circumstances. Once the business context is identified then the *BIE's* can be defined and any number of required qualifications and refinement needed to support the specific business context can be taken into consideration. The *BIELibrary* is used as a container for BIE's. The BIE's exist, by applying restrictions on its underlying core components. This states, that various concepts within the BIE's cannot exist without being defined in its corresponding core component.

6.2.5.2 Step-by-Step modeling guide

1. ***Identify and model Aggregate Business Information Entity.***

   Unlike *ACC's* independent of business context, a UML class stereotyped as *Aggregate Business Information Entity (ABIE)* and created within a *BIELibrary* and is used as a collection of related pieces of information that together carry a distinct meaning of a specific business context. The *ABIE* represents an *ACC* with refined business semantic definition on a specific business context. In other words, an *ABIE* must not contain elements which are not defined in its underlying *ACC*. This is shown as a *basedOn* dependency while modeling. The *BIELibrary* can be composed of zero to multiple *ABIE's*.

   The name of an *ABIE* contains a qualifier representing the business context followed by an underscore character and then the object class term specialized from the core component. The name of the *ABIE* class is equivalent to the object class term defined in CCTS e.g. `AllowanceCharge_Currency Exchange`. The `AllowanceCharge` indicates the qualifier followed by an underscore character, then the `CurrencyExchange` as the object class term.

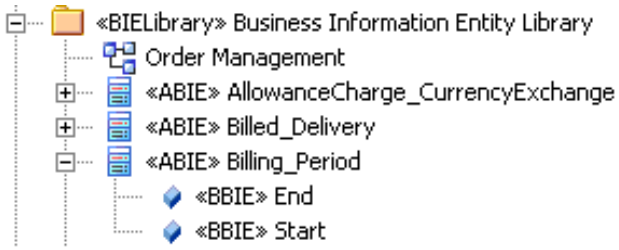

Figure 6.6: Business Information Entity Library - Tree View

2. ***Identify and model Basic Business Information Entity.***

   A single characteristic of an *ABIE* is represented by a property stereotyped as *Basic Business Information Entity (BBIE)* and created beneath its respective *ABIE*. An *ABIE* and is derived from the *ACC* of the *CCLibrary* but with a unique business semantic definition. The name of the *BBIE* attribute is equivalent to the property term defined in CCTS. The *ABIE* can be composed of zero to multiple *BBIE's*.

   In exceptional cases, when an *ABIE* contains more than one *BBIE* with the same property term then the *business data type (BDT)* (*refer section 6.2.6.1*), which symbolizes a representation term is added as a suffix to the property term to ensure

uniqueness to the attribute of the object class in UML modeling. Figure 6.6 shows an example of a *ABIE* `Billing_Period` with a *BBIE's*: `Start` and `End`.

3. ***Identify and model Association Business Information Entity.***

   An *ASBIE* is a BIE that defines a role between a specific *ABIE* (associated aggregate business information entity) associated to another *ABIE* (associating business information entity). It represents a complex property of an (associating) *ABIE* and describes the structure of another (associated) *ABIE*. When an *ASCC* is applied on a business context then a class stereotyped as *Association Business Information Entity (ASBIE)* is created beneath its respective *ABIE*. In other words, the property term of an *ASBIE* is equal to the property term of its underlying *ASCC*. The role name of the association equals the property term as defined in the CCTS. The *ASBIE* has a minimum and a maximum cardinality. *ASBIE* is a UML association of AggregationKind = shared in an information model. Compared to the *ACC*, the *ASBIE* follows similar naming conventions.

6.2.5.3 Tagged values of stereotypes within the BIE Library

The stereotypes *ABIE*, *BBIE* and *ASBIE* inherit from the abstract stereotype *Business Information Entity* thereby inheriting the following tagged values:

- *businessTerm:* A synonym used in business through which the *ASBIE/ABIE/BBIE* are commonly known. An *ASBIE/ABIE/BBIE* may use different business terms as well.
- *definition:* Free text form describing the *ASBIE/ABIE/BBIE* in more detail.
- *dictionaryEntryname:* A unique official dictionary entry name of the *ASBIE/ABIE/BBIE* based on the dictionary entry name rules of the CCTS [31].
- *languageCode:* The code that states which language needs to be used. The Internet Engineering Task Force (IETF) RFC 3066 of January 2001 [51] is used as the source for the code values.
- *uniqueIdentifier:* The unique identifier uniquely represents the *ASBIE/ABIE/BBIE* in the registry. The ITU-T Rec. X.667|ISO/IEC9834-8 Universally Unique Identifier (UUID) [52] scheme is used to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the library to which it belongs.
- *versionIdentifier:* A unique version identifier that represents the version of the *ASBIE/ABIE/BBIE* and which evolves over time over different instanzes.
- *usageRule:* A constraint in a free text format describing the conditions that apply to *ASBIE/ABIE/BBIE*. In order to avoid inconsistency between the usage rule and OCL constraint, a unique identifier should be used for both of them.

Additionally, the *BBIE* and *ASBIE* contains the following tagged value:

- *sequencingKey:* Unique sequencing algorithms may be used by software and storage applications to change the standard defined order of a *BBIE* or *ASBIE* within an *ABIE* respectively. A unique sequencing key is entitled to each *BBIE* or *ASBIE* within an *ABIE* to ensure that a desired order is preserved respectively.

6.2.5.4 Summary of artifacts created in the BIE Library

The *BIELibrary* is a container of BIE's. Results of the *BIELibrary* include

- ***Aggregate Business Information Entities (ABIE)***: A class which derived from the *ACC* for a defined business context.
- ***Basic Business Information Entitie (BBIE)***: A property derived from the *BCC* that stores the business characteristics of the *ABIE*. The *BBIE's* are typified in the *BDT* from the *BDTLibrary*.
- ***Association Business Information Entity (ASBIE)***: An association derived from the *ASCC*, describing the relationship between two *ABIE's*.

6.2.6 Business Data Type Library

6.2.6.1 Overview and purpose

A *BDTLibrary* is used as a container of complex data type elements called *Business Data Types* which restrict their corresponding *CDT's* from the *CCLibrary*. In the *Business Data Type* the main content as well as additional information of a specific element related to a business context is stored.

6.2.6.2 Step-by-Step modeling guide

1. ***Identify and model Business Data Types.***

   If the business document modeler imposes restrictions on the *CDT's* from the *CCLibrary* and uses them in a business context then these complex data types which represent class objects in UML are stereotyped as *Business Data Types (BDT)* and created beneath the *BDTLibrary*. A *CDT* can be used to derive multiple different *BDT's*. A *BDTLibrary* can be composed of zero to multiple *BDT's*. A *BDT* cannot exist without being derived in its corresponding *CDT* by restriction.

2. ***Identify and model the Content and Supplementary Components.***

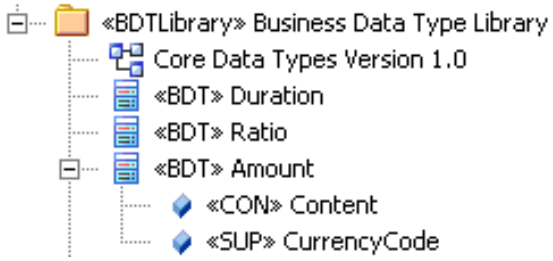

Figure 6.7: Business Data Type Library - Tree View

Similar to the *CON's* and *SUP's* of *CDT's*, the actual information regarding the *BDT* is stored in the property stereotyped as *Content Component (CON)*. Additional information of the *CON* is stored in the property stereotyped as *Supplementary Component (SUP)*. Each *BDT* must contain exactly one *Content Data Type(CON)* and can contain zero to multiple *Supplementary Components(SUP's)*.

The *CON* and *SUP's* of the *BDT's* are typified using the *PRIM's* defined in the *PRIMLibrary, ENUM* or *IDSCHEME* defined in the *ENUMLibrary*. In contrast to the *BDT* name, the *CON* and *SUP's* do not have any qualifiers. However, in order to ensure uniqueness, qualifiers may be applied on *ENUM* types and *IDSCHEME* to denote restrictions. In modeling terms, the *CON's* and *SUP's* represent the attributes to a class *BDT*. Figure 6.7 shows the *BDT* `Amount` containing the *CON* `Content` and additional data in the *SUP* `CurrencyCode`.

6.2.6.3 Tagged values of stereotypes within the BDT Library

The *BDT* as well as the *CON* and *SUP's* generalize from the abstract *business data type property* containing the following tagged values.

- *businessTerm:* A synonym used in business through which the *BDT/CON/SUP* are commonly known. A *BDT/CON/SUP* may use different business terms as well.
- *definition:* Free text description describing the *BDT/CON/SUP* in more detail.
- *dictionaryEntryname:* A unique official dictionary entry name of the *BDT/CON/SUP* based on the dictionary entry name rules of the CCTS [31].
- *languageCode:* The code that states which language needs to be used for the *BDT/CON/SUP* . The Internet Engineering Task Force (IETF) RFC 3066 of January 2001 [51] is used as the source of code values.

- *uniqueIdentifier:* The identifier uniquely represents the *BDT/CON/SUP* in the registry. The ITU-T Rec. X.667|ISO/IEC9834-8 Universally Unique Identifier (UUID) scheme [52] is used to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the library to which it belongs.
- *versionIdentifier:* A unique version identifier representing the version of the *BDT/CON/SUP* and which evolves over time of different instances.
- *usageRule:* A constraint in a free text format describing the conditions that apply to *BDT/CON/SUP*. In order to avoid inconsistency between the usage rule and OCL constraint, a unique identifier should be used for both.
- *modificationAllowedIndicator* An indicator that applies restrictions on enumerations through *BDT's* in order to set the value domain of *CON* and *SUP's*. If the modifications are not desired then the indicator is set as false else it is set as true.
- *enumeration*: Restrict the value domain of a *PRIM* to a set of allowed values.
- *length*: The number of units of length, a value of a given data type must have. The value of a length must be a non-negative integer.
- *pattern*: A constraint in the form of a regular expression on the value space to literals by constraining its lexical space which match a specific pattern.
- *fractionalDigits*: Restricts the value space to those values that can be represented lexically using at most fractional digits to the right of the decimal point. Note that it does not restrict the lexical space directly. A lexical expression that adds additional leading zero digits or trailing fractional zero digits is still permitted
- *maximumExclusive*: An exclusive upper bound of the value space for an ordered data type. A value space is ordered if there exists an order relation defined for that value space. An order relation on a value space is a mathematical relation imposing a total or a partial order on the members of the value space.
- *maximumInclusive*: Inclusive upper bound of the value space for an ordered data type. A value space is ordered if there exists an order relation defined for that value space. An order relation on a value space is a mathematical relation imposing a total or a partial order on the members of the value space.
- *maximumLength*: Maximum number of units of length a value of a given data type shall have. The value of maximumLength must be a non-negative integer.
- *minimumExclusive*: Exclusive lower bound of the value space for an ordered data type. A value space is ordered if there exists an order relation defined for that value space. An order relation on a value space is a mathematical relation imposing a total or a partial order on the members of the value space.

- *minimumInclusive:* Inclusive lower bound of the value space for an ordered data type. A value space is ordered if there exists an order relation defined for that value space. An order relation on a value space is a mathematical relation imposing a total or a partial order on the members of the value space.
- *minimumLength:* Minimum number of units of length, a value of a given data type should have. The value must be a non-negative integer.
- *totalDigits:* Restricts the value space to those values that can be represented lexically using at most total digits. It does not restrict the lexical space directly; a lexical representation that adds additional zero digits or trailing fractional digits is still permitted.

6.2.6.4 Summary of artifacts created within the BDT Library

The *BDTLibrary* is a container of *BDT's*. Results of the *BDTLibrary* include:

- ***Business Data Type***: A class representing a complex data type element. and is based on a *CDT*.
- ***Content Component***: A property where the actual content of the *BDT* is stored and based on the *CON* of the *CDT*.
- ***Supplementary Component***: A property that contains additional meaning to the *CON* of the *BDT* and is based on the *SUP* of the *CDT*.

The *CON's* and as well as the *SUP's* are typified either by using the *PRIM's*, *ENUM's* or *IDESCHEME* from the *PRIMLibrary* and *ENUMLibrary*.

6.2.7 Business Document Library

6.2.7.1 Overview and purpose

The *business document library (DOCLibrary)* is used as a container which contains exactly one business document. The business document is defined by assembling the *ABIE's* which are already created, in a so called *message assembly*. The *information envelope* from the UMM model has exactly one message assembly which serves as a root element of a business document. The *ABIE's* are associated to *message assembly* using *association message assemblies*. For each business document exchanged in a *business transaction* a concrete *DOCLibrary* is created.

6.2.7.2 Step-by-Step modeling guide

1. ***Identify and model Message Assembly.***

   Several *BIE's* are aggregated and a final business document is created and stereotyped as a *Message Assembly (MA)*. At least one *message assembly* must be defined within the *DOCLibrary*. It is connected to the *InformationEnvelope* of the UMM model to represent a business message. The *message assembly* serves as a root element of a business document. It has an optional *standard business document header* derived from the UN/CEFACT's Standard Business Document Header Specification which serves the identification of document type, technical sender, receiver etc.

2. ***Design the Association Message Assembly.***

   If *MA's* are related to one another then they are aggregated using an association stereotyped as *Association Message assembly (ASMA)* to a specific business document. A *MA* can also be aggregated to *BIE's* from the *BIELibrary* through an *ASMA*. The source of the *ASMA* must be a *MA*. The target can be associated either with a *MA* or with as *ABIE*. A *MA* can be used to aggregate one to multiple *ABIE's*. Furthermore, the optional *standard business document header* is also associated to the root *MA* through an *ASMA*.

   Figure 6.8 shows hows the *Message Assembly* `OrderEnvelope` is associated to the *ABIE's* using *Association Message assemblies*. Furthermore, using the drag and drop functionality, the *InfEnvelope* `OrderAcceptanceEnvelope` from the *business information view* of the UMM model is associated with the *MA* `OrderEnvelope`. Additionally, the *standard business document header* from the SBDH library is also used in the UPCC model using the drag and drop functionality.

6.2.7.3 Tagged values of stereotypes within the DOC Library

The stereotype *MA* contains the following tagged values:

- *versionIdentifier:* A unique version identifier that represents the version of the *MA* and evolves over time of different instances.
- *minorVersionIdentifier:* A series of minor changes within the major version of the *MA* represented by the *versionIdentifier*.
- *revisionIdentifier:* Represents a revision version in addition to the *versionIdentifier* and *minorVersionIdentifier* of the *MA*.

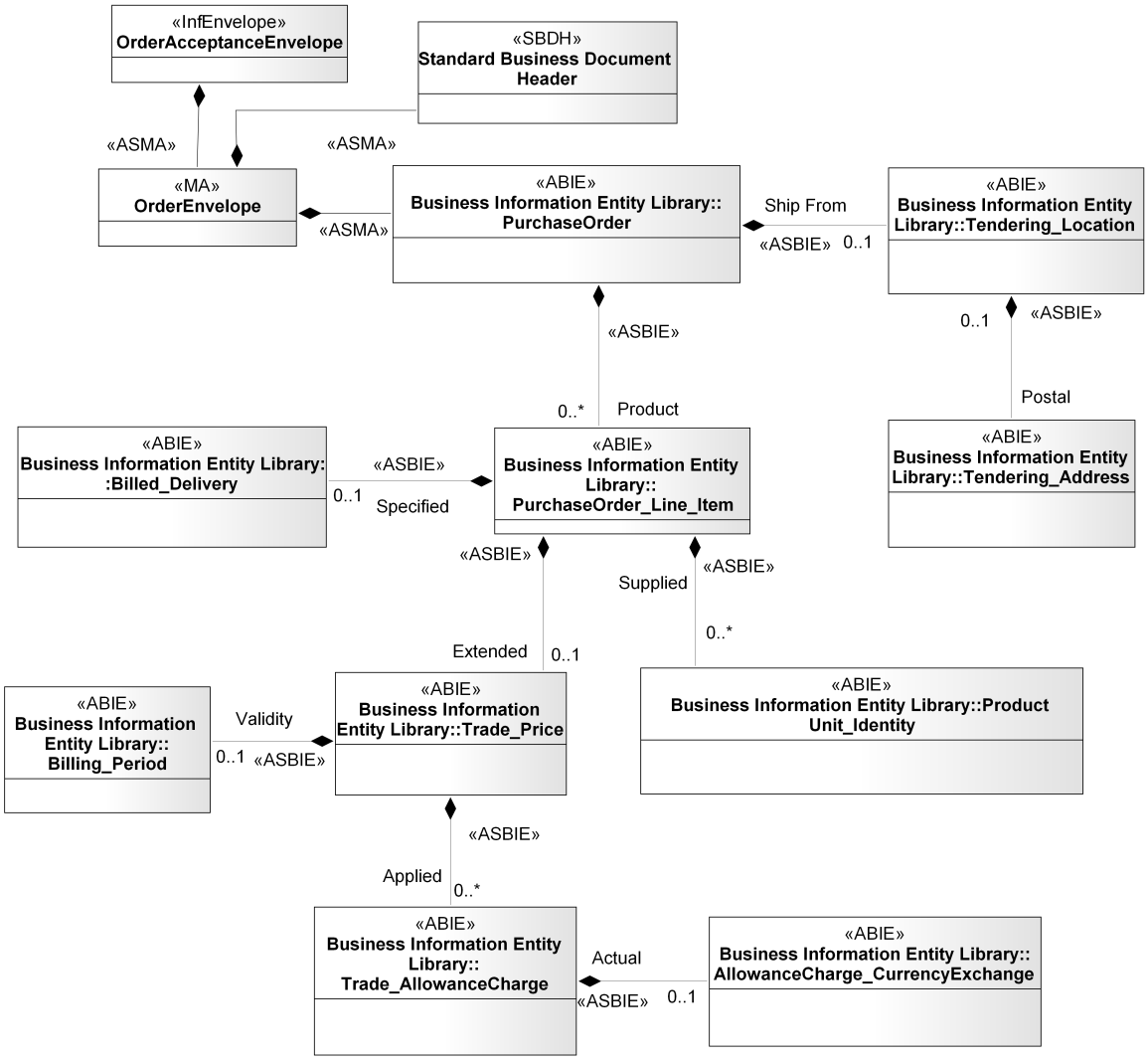

Figure 6.8: Business Document Library - Quote Envelope

6.2.7.4 Summary of artifacts created within the DOC Library

Results of the *DOC Library* include the following artifacts:

- ***Message assembly***: This class aggregates several *ABIE's* in order to define a business message.
- ***Association Message Assembly***: An association between *MA's* and *MA's* or *ABIE's*

6.2.7.4.1 Summary of artifacts created within the UPCC Library

The *UPCCLibrary* results in seven library packages in order to create a business document model. These include:

- ***Core Component Library***: A container of core components which include: *ACC's, ASCC's* and *BCC's*.
- ***Core Data Type Library***: A container of *CDT's* representing the data type elements of *core components*.
- ***Primitive Type Library*** : A container of *PRIM's* which are already defined in the Core Component Data Type Catalogue 3.0.
- ***Enumeration Type Library***: A container of *ENUM's* and *IDSCHEME's* which restrict the *PRIM's* to a well defined set of values.
- ***Business Information Entity Library***: A container of following *BIE's*: *ABIE's*, *ASBIE's* and *BBIE's*.
- ***Business Data Type Library***: A container of *BDT's* representing the data type elements of *BIE's*.
- ***Business Document Library***: A container of exactly one business document that is used during the exchange of a *business transaction* in UMM.

CHAPTER 7

Conclusion

UN/CEFACT's Modeling Methodology and UML Profile for Core Components have evolved from the traditional EDI-based technologies. By providing a standard means for modeling business processes and designing business documents, UMM enables modeling collaborative B2B processes and UPCC encourages standard business information exchange between trading partners. Since UMM has been in existence for some time now, it has proven its value and support in business process collaborations. Applied works of UMM are seen for example in the field of a rural fisher community [71] and in an international e-Infrastructure (eIS) project [72]. The concepts of UPCC are for example illustrated in the management of hybrid cloud deployments [73].

Building on the first edition of *A UML Profile and Add-In for UN/CEFACT's Modeling Methodology* based on UMM 1.0, *A User Guide for UN/CEFACT's Process and Data Modeling Methodologies* based on UMM 2.0, is fully updated and is as comprehensive as possible to incorporate the latest in UMM methodology. This thesis serves as a concise guide with design guidelines and worksheets. It outlines the steps involved in the creation of artifacts and different views required to design a framework for a UMM model with a real-world scenario. This thesis is useful when UMM appears to be complex, making a modeller unaware of the steps required to create a business process model. With the help of this guide, the user will have a firm grasp of the fundamental concepts, competitive advantages and potential best uses of the UMM methodology.

Although UMM is a model-driven methodology, it is still lacking in a few issues. Limitations of the UMM are ignoring the so called "middle-out" approach while focusing on the top-down and bottom-up approach. While the top-down approach studies the requirements suitable to build a UMM model, the bottom-up approach concentrates on the collaborative environment between trading partners and the middle-out approach focuses on the internal business processes from the trading partner perspective. However, enterprises would find it expensive to re-organize their internal business processes

based on the collaborative business processes or even to create new internal business processes while having existing business processes. This requires a suitable method to capture internal business processes and integrate them with existing and new business processes. Secondly, UMM follows on a process-centric approach. However, for a successful realization of an inter-organizational business collaboration, a combination of process-centric as well as document-centric approaches needs to be taken into consideration. Thirdly, while UMM is a choreography language used for modeling collaborative business processes, it comes to its limitations when trying to orchestrate services. A different methodology is required for modeling internal business process as a basis for service orchestration and they can follow a complex pattern compared to UMM choreography [65]. In [70], the author reviews that UMM focuses more on the low-level message exchange leading to rigid models compared to the commitment-based business modeling methodology: Comma.

There are a few drawbacks in UPCC. These include the meta-model restrictions of UML through the UML profile in order to achieve certain requirements in business document modeling. Since the representation of a few concepts of CCTS in the meta-model are interrupted, the modeler needs to accept the constraints required for modeling in UPCC. Secondly, for every release of core component standard, UN/CEFACT needs to update a new version of UPCC in order to meet the requirements of the new core component version. Thirdly, the lack of tool support is another significant drawback for a wide acceptance of core component technology in various applications. Electronic core component registry is also found to be crucial for the entire core component field of business document design [75] .

There are many points that are still open and subject to further research within the topics of UMM and UPCC. Future work in UMM will focus on investigating the integration of value-based requirements engineering into UMM. In value-based requirements engineering, the focus is on business models in order to evaluate the economic justification for e-business systems. Prominent approaches for analysing business models are e3-value methodology [67], the Resource-Event-Agent (REA) theory [68] or the Business Model Ontology [69]. Another open issue, is the mapping of business collaborations that design the flow of business transactions to BPEL [65]. Furthermore, future work concentrates on capturing significant information from attributes of the REA's *economic commitments* and concentrating on the REA's state driven approach for mapping to UMM. REA delivers the requirements from an economic point of view and serves as an input for modeling UMM compliant business process model [63] [64].

Future works in UPCC explores the development and integration of matching tools, given the opportunity to match business document standards automatically. Furthermore, other open research includes taking large models into consideration in order to ensure proper usability of business document modeling tools.

APPENDIX A

Worksheets

Business Domain View	
General	
Name	Name of the *business domain view*
Description	Free text information describing the business domain in more detail
Business Library Information	
URI	A unique identifier that represents the *business library* package in the registry. A UUID (Universally Unique Identifier) is used within the URI structure scheme to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library*.
BusinessTerm	A synonym frequently used for the *business domain view* in business
Version	The current version of the *business library* in the registry. The version information should not be assigned by the modeler rather must be managed in the registry. An unregistered library must not have any version assigned.
Status	It represents the current lifecycle status of the *business domain view* registered in the *business library* package. The status is set by the registry. The possible status representing the *business domain view* are *proposed, approved, mandatory, validated* and *implemented*. An registry unregistered must not have any status information assigned.
Owner	The owner of the *business library* which could be an organization, institution or an individual.
Copyright	Copyright of the *business library* package.
Reference(s)	Contains location information to additional resources where further information about the *business library* could be found.
Business Area(s) (add columns as needed)	
Business Area	A list of *business areas* within the scope of the *business domain view*.

Table A.1: Business Domain View Worksheet

Business Area	
General	
Name	Name of the *business area/business category*
Description	Free text information describing the *business area* in more detail
Details	
Objective	The goal intended to be attained by the *business area* or the *business category* under consideration.
Scope	Defines the limit of access of this *business area* within the business collaboration model and is defined as public, private or protected
Business Opportunity	A strategic interest in this *business area* from the business perspective.
Included in	Name of parent *business area/business category* or the business domain
Business Library Information	
URI	A unique identifier that represents the *business library* package in the registry. A UUID (Universally Unique Identifier) is used within the URI structure scheme to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library*.
BusinessTerm	A synonym frequently used for this *business area/business category* in business.
Version	The unique identifier that represents the version of the *business library* in the registry. It represents its structure in the UUID(Universally unique Identifier) scheme. However it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library*.
Status	It represents the current lifecycle status of the *business area/business category* registered in the *business library* package. The status is managed in the registry. The possible status representing the *business area/business category* in UMM are *proposed, approved, mandatory, validated* and *implemented*. A registry unregistered must not have any status information assigned.
Owner	The owner of the *business library* which could be an organization, institution or an individual
Copyright	Holds copyright of the *business library* package.
Reference(s)	Holds location information to additional resources where further information about the *business library* could be found.
Business Area(s): (insert additional nested business areas if appropriate; otherwise fill process areas that apply)	
Business Area No 1..n	List of *business areas/business category* within the scope of this *business area/business category*.
Process Area(s): (insert additional nested process areas for this business area if no other business areas are found.)	
Process Area No 1..n	List of *process areas* within the scope of this *business area/business category*.

Table A.2: Business Area Worksheet

Process Area	
General	
Name	Name of the *process area*
Description	Free text information describing the *process area* in more detail
Details	
Objective	The goal intended to be attained by the *process area* under consideration.
Scope	Defines the limit of access of this *process area* within the business collaboration model and is defined as public, private or protected
Business Opportunity	A strategic interest from a business perspective in this *process area*
Included in	Name of the parent *business area* or *process area*
Business Library Information	
URI	A unique identifier that represents the *business library* package in the registry. A UUID (Universally Unique Identifier) is used within the URI structure scheme to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library*.
BusinessTerm	A synonym frequently used in business for the *process area*
Version	The unique identifier that represents the version of the *business library* in the registry. It represents its structure in the UUID(Universally Unique Identifier) scheme. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library*.
Status	It represents the current lifecycle status of the *process area* registered in the *business library* package. The status is managed in the registry. The possible status representing the *process area* in UMM are *proposed, approved, mandatory, validated* and *implemented*
Owner	The owner of the *business library* which could be an organization, institution or an individual
Copyright	Holds copyright of the *business library* package.
Reference(s)	Holds location information to additional resources where further information about the *business library* could be found.
Process Area(s) (if present or add additional process areas if needed)	
Process Area No 1..n	List of child (top level) *process areas* within the scope of a *process area*.

Table A.3: Process Area Worksheet

Business Process	
General	
Name	Name of a *business process*
Description	Free text information describing the *business process* in more detail
Details	
Classified to Business Areas and Process Areas	List all the ancestor *business areas* and *process areas* to which this *business process* is categorized.
Participants and their interests	Parties involved in the execution of the *business processes* are listed along with their areas of interest.
Stakeholders and their interests	Parties who take interest in a *business process* but are not active during the execution of any *business process*. Their areas of interest are also listed here.
Reference(s)	Holds location information to additional resources where further information about the *business library* could be found.
Start/End Characteristics	
Pre-condition	A set of conditions that need to be fulfilled before the execution of a *business process*
Post-condition	A set of conditions that need to be fulfilled after the execution of a *business process*
Begins When	Specifies certain business events that trigger the initiation of a *business process*. It could also be a case where a certain state needs to be specified in order to begin a *business process*.
Ends When	Specifies a list of business events or conditions that lead to the termination of a *business process*.
Actions	Describes the activities performed during the execution of a *business process*
Exceptions	Lists all exception conditions that that causes the *business process* to terminate before its normal completion
Relationships	
Included Business Processes	*Business processes* that take part in the execution of the current *business process*.
Affected Business Entities	Identifies *business entities* affected during the execution of a *business process*.

Table A.4: Business Process Worksheet

Business Entity	
General	
Business Entity Name	Name of the *business entity*
Description	More information on the business entity is detailed here
Business Library Information	
URI	A unique identifier that represents the *business library* package in the registry. A UUID (Universally Unique Identifier) is used within the URI structure scheme to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library*.
BusinessTerm	A synonym frequently used in business for the *business entity*
Version	The unique identifier that represents the version of the *business library* in the registry. It represents its structure in the UUID (Universally Unique Identifier) scheme. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the business library.
Status	Represents the current lifecycle status of the *business entity* registered in the business library package. The status is managed in the registry. The possible status representing the *business entity* in UMM are *proposed, approved, mandatory, validated* and *implemented*
Owner	The owner of the *business library* which could be an organization, institution or an individual
Copyright	Holds copyright of the *business library* package.
Reference(s)	Holds location information to additional resources where further information about the *business library* could be found.
Lifecycle	
Pre-Condition	A set of conditions that need to be fulfilled before the execution of the *business entity*
Post-Condition	A set of conditions that need to be fulfilled after the execution of the *business entity*
Begins When	Business events that trigger the initiation of the *business entity*. It could also be used where a certain state needs to be specified to begin the process.
Ends When	Business events or conditions that lead to the termination of the *business entity*.
Exceptions	Lists all exception conditions that that causes the *business entity* lifecycle to terminate before its normal completion
Lifecycle States (add more Business Entity States if needed)	
Business Entity State	
Name	Name of the *business entity state*.
Description	A free text explanation of the *business entity state*
Preceding State(s) including events and transition conditions	Identifies the predecessors along with the events and condition that have occurred before reaching this specific state
Valid Actions	Business actions performed once the *business entity* reaches a specific state.

Table A.5: Business Entity Worksheet

Business Transaction Use Case	
General	
Name	Name of the *business transaction use case*
Description	Free text information of the *business transaction use case*
Business Library Information	
URI	A unique identifier that represents the *business library* package in the registry. A UUID (Universally Unique Identifier) is used within the URI structure scheme to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library*.
BusinessTerm	A synonym frequently used in business for the *business transaction use case*
Version	A unique identifier that represents the version of the *business library* in the registry. It represents its structure in the UUID (Universally Unique Identifier) scheme. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the business library.
Status	It represents the current lifecycle status of the *business transaction use case* registered in the *business library* package. The status is managed in the registry. The possible status representing the *business transaction use case* in UMM are *proposed, approved, mandatory, validated* and *implemented*
Owner	The owner of the *business library* which could be an organization, institution or an individual
Copyright	Holds copyright of the *business library* package.
Reference(s)	Holds location information to additional resources where further information about the *business library* could be found.
Details	
Requesting Role	Determines a specific role that initiates the *business transaction* to take place.
Responding Role	Identifies the role that takes action by responding back to the *business transaction* that was initiated.
Requesting Activity	Determines the activity performed by the requesting role that provokes the *business transaction* to take place.
Responding Activity	Specifies the activity initiated by the responding role as a response to the *business transaction* that was already initiated.
Is Included In (Name of Business Collaboration)	Specifies a list of *business collaboration use cases* to which this *business transaction use case* is included in.
Start/End Characteristics	
Affected Business Entities	States the names of the *business entities* whose *business entity states* are affected due to the execution of the business transaction.
Pre-condition	A set of conditions that need to be fulfilled before the execution of the *business transaction*
Post-condition	A set of conditions that need to be fulfilled after the execution of the *business transaction*

Begins When	Specifies certain business events that trigger the initiation of the business transaction. It also specifies a certain state that the *business process* needs to reach to initiate the *business transaction*.
Ends When	Specifies certain business events that lead to the termination of a *business transaction* or a certain state which terminates the transaction
Exceptions	Identifies the errors that occur during the execution of the *business transaction*.

Table A.6: Business Transaction Use Case Worksheet

Business Transaction	
General	
Name	Name of the *business transaction*
Description	Free text information describing the *business transaction* in more detail
Business Library Information	
URI	A unique identifier that represents the *business library* package in the registry. A UUID (Universally Unique Identifier) is used within the URI structure scheme to ensure uniqueness. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library*.
BusinessTerm	A synonym frequently used in business for the *business transaction*
Version	The unique identifier that represents the version of the *business library* in the registry. It represents its structure in the UUID (Universally Unique Identifier) scheme. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the *business library*.
Status	It represents the current lifecycle status of the *business transaction* registered in the *business library* package. The status is managed in the registry. The possible status representing the business transaction in UMM are *proposed, approved, mandatory, validated* and *implemented*
Owner	The owner of the *business library* which could be an organization, Institution or an individual
Copyright	Holds copyright of the *business library* package.
Reference(s)	Holds location information to additional resources where further information about the *business library* could be found.
Details	
Select Business Transaction Pattern	There are namely six different kinds of business patterns used in the business transaction. - Information Distribution - Request/Response - Request/Confirm - Query/Response - Notification - Commercial Transaction

Secure Transport	Is set to true if the business information is exchanged via a secure transport channel.
Requestor's Side	
Requesting Role	Identifies the initiator who initiates the *business transaction*.
Requesting Business Action Name	Name of business action executed by the requesting *authorized role*.
Time to Respond	Specifies the period of time required by the responding role to reply in a two-way *business transaction*. In case of a one-way *business transaction*, the value is set to null.
Time to Acknowledge Receipt	Specifies the period of time required by the responding role to acknowledge the receipt of business information sent from the requesting role. If no specific time period is required then the value here is set to null.
Time to Acknowledge Processing	It is the duration of time, the requestor sends the business information to the responder until the time an acknowledgement of processing is received back by the requestor. The value is set to null if not specific time duration needs to be allotted.
Authorization Required	Is set as true if the responding role must authorize itself.
Non Repudiation Required	True if the responding role must not repudiate the execution of the business action which inputs and outputs business information.
Non Repudiation of Receipt Required	Is true if an acknowledgement of receipt is required from the responder.
Intelligible Check Required	Is set as true if the responding role must check that the business information is not garbled during transmission, before an acknowledgement of receipt is sent back to the requestor.
Number of Retries	Specifies the number of times, the requesting *authorized role* needs to re-initiate a *business transaction* in case of a time-out exception like exceeding the time to acknowledge receipt, time to acknowledge processing or the time to respond.
Responder's Side	
Responding Role	Identifies the responder which replies back to the transaction that took place.
Responding Business Action Name	Specifies the name of the business action, executed by the responding role
Time to Acknowledge Receipt	Specifies the period of time required by the requesting role to acknowledge the receipt of business information from the responding role. If no specific time period is required then the value here is set to null
Time to Acknowledge Processing	The duration of time, the responder sends the business information to the requestor until the time an acknowledgement of processing is received back by the responder.
Authorization Required	Is true if the requesting role needs to be authorized. The requesting role must sign the business document exchanged and the responding role must validate the business authorization and accept the business transaction with the requesting role.
Non Repudiation Required	This tag is specified as true if the requesting role must not repudiate the execution of the business action which inputs and outputs business information.

Non Repudiation of Receipt Required	Is true if an acknowledgement of receipt is required from the requestor's side. Non repudiation of receipt ensures that the requesting party must not be able to deny sending the acknowledgement of receipt.
Intelligible Check Required	True, if the requesting role must check that the business information is not garbled during transmission before an acknowledgement of receipt is sent back to the responder.
Business Information Envelopes	
Requesting Information Envelope	
Name	Name of the requesting information envelope
Are Contents Confidential?	Is true if the information is encrypted so that the unauthorized parties cannot view the information.
Is the Envelope Tamper-proof?	Is true if the business document exchanged is digitally signed by the requestor.
Authentication Required?	Is true if a requestor's digital certificate is linked with the document which provides a proof of signer's identity.
Responding Information Envelope (add more Responding Information Envelopes if different response documents are possible)	
Name	Name of the responding information envelope.
Resulting Business Entity State (including transition condition)	States the name of the resulting *business entity state* along with the transition condition occurred due to the business transaction.
Are Contents Confidential?	Is true if the information is encrypted so that the unauthorized parties cannot view the information.
Is the Envelope Tamper-proof?	Is true if the business document exchanged is digitally signed by the responder.
Authentication Required?	Is true if a responder's digital certificate is linked with the document which provides a proof of signer's identity

Table A.7: Business Transaction Worksheet

Business Collaboration Use Case	
General	
Name	Name of the *business collaboration use case*
Description	Free text describing the *business collaboration use case* in more detail
Business Library Information	
URI	The unique identifier representing the *business library* package in the registry. It represents its structure in the UUID (Universally Unique Identifier) scheme. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the business library.
BusinessTerm	A synonym used in business through which the *business collaboration use case* is commonly known.

Version	The unique identifier that represents the version of the *business library* in the registry. It represents its structure in the UUID (Universally Unique Identifier) scheme. However, it can be chosen freely in any other structure scheme as long as it guarantees uniqueness in the business library.
Status	It represents the current lifecycle status of the *business collaboration use case* registered in the *business library* package. The status is managed in the registry. The possible status representing the *business collaboration use case* in UMM are *proposed, approved, mandatory, validated* and *implemented*
Owner	The owner of the *business library*, which could be an organization, institution or an individual
Copyright	Holds copyright of the *business library* package.
Reference(s)	Holds location information to additional resources where further information about the *business library* could be found.
Participants	
Participating Role [add more participating roles in case of a multi-party collaboration]	Lists the *business partners* that participate in the business collaboration. Minimum two *authorized roles* take part in the *business collaboration use case*.
Is Included In (Name of parent Business Collaboration – if there is any)	List the names of the additional *business collaboration use cases* in which this business collaboration is included.
Start/End Characteristics	
Affected Business Entities	A set of conditions that need to be fulfilled before the execution of the *business collaboration*
Pre-condition	A set of conditions that need to be fulfilled after the execution of the *business collaboration*
Post-condition	Specifies certain business events that trigger the initiation of the business collaboration. It could also be used where a certain state needs to be specified to begin the process.
Begins When	Specifies the business events that lead to the termination of the business collaboration.
Ends When	Describes the activities that are performed during the execution of the business collaboration
Exceptions	Identifies the errors that may occur during the execution of the business collaboration.
Included Business Transaction Use Cases (add more Business Transaction Use Cases if needed)	
Business Transaction Use Case Name	Lists the names of the *business transaction use cases* that are included in a business collaboration.

Table A.8: Business Collaboration Use Case Worksheet

Business Collaboration Protocol	
General	
Name	Name of the *business collaboration protocol*
Description	A free text information describing the *business collaboration protocol* in more detail.
Participants (copy from Business Collaboration Use Case Worksheet)	
Participating Role [add more participating roles if elicited in the Business Collaboration Use Case Worksheet]	Identifies the *authorized roles* that participate in a *business collaboration*. At least two *authorized roles* take part in the business collaboration. The participants of *business collaboration protocol* are identified from the *business collaboration use case*.
Included Business Transaction Actions / Business Collaboration Actions[add more Business transaction actions if needed]	
Business Transaction Action	
Name	Name of the *business transaction* executed through the *business transaction call*.
Preceding Action(s) including transition condition	The business activities that occur before the *business transaction call* is executed. The transition conditions that initiate the *business transaction call* are also stated here.
Initiating Role	Name of the *authorized role* that initiates the business transaction through the *business transaction call*. This *authorized role* must in listed in the participating role above.
Reacting Role	Name of the *authorized role* that responds to the business transaction through the *business transaction call*. This *authorized role* must be listed in the participant role list above.
Business Collaboration Action [delete if not required or add more if needed]	
Name	Name of the additional *business collaboration use case* included in a *business collaboration*
Preceding Action(s) including transition condition	The business actions executed before the business collaboration action is called are listed here. The transition conditions that initiate the *business collaboration call* are also listed here.
Role Mapping [add more role mappings if required]	• ***Role in this Business Collaboration***: States the task played by the additional business collaboration. • ***Role in the nested Business Collaboration***: Describes the job played by the nested *business collaboration use case* in a business collaboration.

Table A.9: Business Collaboration Protocol Worksheet

APPENDIX B

Stereotype Abbreviations - UMM

STEREOTYPE	BASE CLASS	PARENT	ABBREVIATION
AuthorisedRole	Actor	N/A	AuthorizedRole
bArea	Package	Business Category	BusinessArea
bCategory	Package	Business Library	BusinessCategory
bChoreographyV	Package	BusinessLibrary	BusinessChoreographyView
bCollaborationCall	CallBehaiviorAction	N/A	BusinessCollaborationCall
bCollaborationUC	UseCase	bProcessUC	BusinessCollaborationUseCase
bCollaborationProtocol	Activity	N/A	BusinessCollboratonProtocol
bCollaborationV	Package	BusinessLibrary	BusinessCollaborationView
bCollModel	Package	Business Library	BusinessCollaborationModel
bCPartition	ActivityPartition	N/A	BusinessCollaborationPartition
bDataV	Package	BusinessLibrary	BusinessDataView
bDomainV	Package	Business Library	BusinessDomainView

bEInternalState	ObjectNode	N/A	InternalBusinessEntityState
bEntity	Class	N/A	BusinessEntity
bEntityV	Package	Business Library	BusinessEntityView
bEState	State	N/A	BusinessEntityState
bESharedState	ObjectNode	N/A	SharedBusinessEntityState
bInformation	Class	N/A	BusinessInformation
bInformationV	Package	BusinessLibrary	BusinessInformationView
bLibrary	Package	N/A	BusinessLibrary
bPartner	Actor	Stakeholder	BusinessPartner
bPartnerV	Package	Business Library	BusinessPartnerView
bProcess	Activity	N/A	BusinessProcess
bProcessAction	Action	N/A	BusinessProcessAction
bProcessUC	UseCase	N/A	BusinessProcessUseCase
bRealisationUC	UseCase	N/A	BusinessRealizationUseCase
bRealisationV	Package	BusinessLibrary	BusinessRealizationView
bRequirementsV	Package	BusinessLibrary	BusinessRequirementsView
bTPartition	Partition	N/A	BusinessTransactionPartition
bTransaction	Activity	N/A	BusinessTransaction
bTransactionCall	CallBehaviorAction	N/A	BusinessTransactionCall
bTransactionUC	UseCase	bProcessUC	BusinessTransactionUseCase
bTransactionV	Package	BusinessLibrary	BusinessTransactionView
BusinessAction	Action	N/A	BusinessAction
InfEnvelope	Class	BusinessInformation	InformationEnvelope
InfPin	Pin	N/A	InformationPin
initFlow	Information Flow	N/A	InitiatingFlow
isOfInterestTo	Dependency	N/A	isOfInterestTo
mapsTo	Dependency	N/A	mapsTo

NestedCollaboration	CallBehaiviorAction	N/A	NestedBusinessCollaboration
Participates	Association	N/A	participates
ProcessArea	Package	Business Category	ProcessArea
reflow	Information Flow	N/A	RespondingFlow
ReqAction	Action	BusinessAction	RequestingBusinessAction
ResAction	Action	BusinessAction	RespondingBusinessAction
ResInfPin	Pin	InformationPin	RespondingInformationPin
ReqInfPin	Pin	InformationPin	RequestingInformationPin
Stakeholder	Actor	N/A	Stakeholder

APPENDIX C

Stereotype Abbreviations - UPCC

STEREOTYPE	BASE CLASS	PARENT	ABBREVIATION
ABIE	Class	-	Aggregate Business Information Entity
ACC	Class	-	Aggregate Core Component
ASBIE	Association	N/A	Association Business Information Entity
ASCC	Association	N/A	Association Core Component
ASMA	Association	N/A	Association Message Assembly
basedOn	Dependency	N/A	basedOn
BBIE	Property	N/A	Basic Business Information Entity
BCC	Property	N/A	Basic Core Component

BDT	Class	N/A	Business Data Type
BDTProperty	Property	N/A	Business Data Type Property
BDTLibrary	Package	Business Library	Business Data Type Library
BIE	N/A	N/A	Business Information Entity
BIELibrary	Package	Business Library	Business Information Entity Library
bLibrary	Package	N/A	Business library
BusinessContext	Class	-	Business Context
BusinessProcessContextValue	Class	-	Business Process Context Value
BusinessProcessRoleContextValue	Class	ContextValue	Business Process Role Context Value
GeopoliticalContextValue	Class	ContextValue	Geopolitical Context Value
CCLibrary	Package	Business Library	Core Components Library
CCTS_DT_3p0	Enumeration	N/A	CCTS_DT_3p0
CDT	Class	N/A	Core Data Type
CDTProperty	Property	N/A	Core Data Type Property
CDTLibrary	Package	Business Library	Core Data Type Library
ClassificationScheme	Class	N/A	Classification Scheme
CodeListEntry	EnumerationLiteral	N/A	Code List Entry
CON	Property	N/A	Content Component
ContextValue	Class	N/A	Context Value
DOCLibrary	Package	Business Library	Business Document Library
ENUM	Enumeration	N/A	Enumeration Type
ENUMLibrary	Package	Business Library	Enumeration Library
equivalentTo	Dependency	N/A	equivalentTo
IDSCHEME	DataType	N/A	Identifier Scheme Type

IndustryClassificationContextValue	Class	ContextValue	Industry Classification Context Value
isbasedOn	Dependency	N/A	Is Based On
isEquivalentTo	Dependency	N/A	Is Equivalent To
MA	Class	-	Message Assembly
OfficialContraintsContextValue	Class	ContextValue	Official Constrainsts Context Value
PRIM	Data Type	N/A	Primitive Type
PRIMLibrary	Package	Business Library	Primitive Data type Library
ProductClassificationContextValue	Class	ContextValue	Product Classification Context Value
SUP	Property	N/A	Supplimentary Component
SupportingRoleContextValue	Class	ContextValue	Supporting Role Context Value
SystemCapabilitiesContextValue	Class	ContextValue	System Capabilities Context Value
UPCCLibrary	Package	Business Library	UPCC Library
UsageRule	Class	N/A	Usage Rule
ValueDomain	N/A	N/A	Value Domain

Bibliography

[1] Rosen A. *The E-commerce Question and Answer Book*, 2nd edition, pp.1-8, 2002.

[2] Schatz W. *EDI: Putting the muscle in commerce and industry.* Scott, Foresman & Co., pp.104-112, 1990.

[3] ANSI ASC. *ANSI ASC X12*, `http://www.x12.org`, 1983, Last revisited: Dec 2012.

[4] UN/CEFACT. *United Nations Electronic Data Interchange For Administration, Commerce and Transport (UN/EDIFACT)* `http://www.unece.org/trade/untdid/welcome.htm`, 1988, Last revisited: Feb 2013.

[5] J. Berge. *The EDIFACT Standards*, 2nd edition, Blackwell Publishers, Cambridge, MA, USA, 1994.

[6] ISO. *Open-edi Reference Model,* ISO/IEC JTC 1/SC 30 ISO Standard 14662, Second Edition, 2004.

[7] UN/CEFACT. *UN/CEFACT's Modeling Methodology (UMM), UMM Meta Model - Foundation Module*, Technical Technical Specification V1.0, `http://www.unece.org/cefact/umm/UMM_Foundation_Module.pdf`, March 2006, Last revisited: Feb 2013.

[8] UN/CEFACT Technologies and Methodologies Group (TMG). *UN/CEFACT's Modeling Methodology (UMM), UMM Meta Model - Foundation Module*, Version 2.0 Technical Specification, `http://www.untmg.org/umm/spec/foundation/2_0`, May 2012, Last revisited: Feb 2013.

[9] Pilone D., Pitman N. *UMM 2.0 in a Nutshell, A Desktop Quick Reference*, 2005.

[10] Miles R., Hamilton K. *Learning UML 2.0, A Pragmatic Introduction to UML*, 2006.

[11] Bussler C. *Enterprise Application Integration and Business-to-Business Integration Processes*. In: Process Aware Information Systems: Bridging People and Software Through Process Technology, pp. 61-82, Wiley Publishing, 2005.

[12] van der Aalst, W.M.P.s, Weske, M. *The P2P approach to Interorganizational workflows.* In Dittrich, K.R., Geppert, A., Norrie, M.C., eds.: CAiSE. Volume 2068 of Lecture Notes in Computer Science., pp.140-156, 2001.

[13] Peltz C. *Web Services Orchestration and Choreography.* IEEE Computer 36, pp. 46-52, 2003.

[14] Hofreiter B. *Registering UML Models for Global and Local Choreographies*, in ACM International Conference Proceeding Series, 342, 2008.

[15] OMG Object Management Group. *Meta-Object Facility*, Version 2.0 Specification, 2006.

[16] HongXingLiu, YanShengLu, QingYang. *XML conceptual modeling with XUML*, In Proc. of ICSE, 2006.

[17] Combi C., Oliboni B. *Conceptual modeling of XML data*, in SAC: Proceedings of the 2006 ACM symposium on Applied computing, ACM, New York, NY, USA, pp. 467-473, 2006.

[18] Pagano D., Brüggemann-Klein A. *Engineering Document Applications - From UML Models to XML Schemas,* Montréal, Canada, In Proceedings of Balisage: The Markup Conference 2009. Balisage Series on Markup Technologies, vol. 3, 2009.

[19] Necaský M. *Reverse Engineering of XML Schemas to Conceptual Diagrams*, Proceedings of The Sixth Asia-Pacific Conference on Conceptual Modelling. CRPIT, 96. Australian Computer Society. pp. 117-128, Wellington, New Zealand 2009.

[20] France R.B., Ghosh S., Dinh-Trong T., Solberg A. *Model-Driven Development Using UML 2.0: Promises and Pitfalls*, Computer, Vol. 39, No. 2., pp. 59-66. 2006.

[21] Liegl P., Mayrhofer D. *A Domain Specific Language for UN/CEFACT's Core Components*, 2009.

[22] OASIS. *Universal Business Language 2.0 (UBL)*, `http://docs.oasis-open.org/ubl/os-UBL-2.0/UBL-2.0.html`, 2006, Last revisited: Feb 2013.

[23] Zachman J.A. *John Zachman's Concise Definition of the The Zachman Framework,* Zachman International, 2008.

[24] TOGAF. http://www.opengroup.org/architecture/togaf9-doc/arch/, Last revisited: Feb 2013.

[25] van't Wout, J., Waage, M., Hartman, H., Stahlecker, M., Hofman, A. *The Integrated Architecture Framework Explained*, 1st Edition., 2010.

[26] Information Framework. http://www.evernden.net/, Last revisited: Feb 2013.

[27] Yan Y., Klein M. *Web Services vs. ebXML: An Evaluation of Web Services and ebXML for e-Business Applications*, 2007.

[28] Chiu E. *ebXML Simplified: A Guide to the New Standard for Global E Commerce*, 2002.

[29] Naujok K., Huemer C. *Designing ebXML - The Work of UN/CEFACT*; in:Ontologies-Based Business Integration, Springer, ISBN: 978-3-540-75229-5, 79 - 93, Heidelberg, 2008.

[30] Object Management Group (OMG). *Business Process Modeling Notation Specification*, January 2011. Version 2.0, http://www.omg.org/spec/BPMN/2.0/PDF. Last revisited: Feb 2013.

[31] UN/CEFACT. Core Components Technical Specification 3.0 (CCTS), http://www.untmg.org/ccts/spec/3_0, 2009, Last revisited: Feb 2013.

[32] ANSI ASC. Context Inspired Component Architecture (CICA), http://www.disa.org/x12org/meetings/x12trimt/cica.cfm, 2002, Last revisited: Feb 2013.

[33] UN/CEFACT. *User Guide for CCTS 3 and NDR 3*, http://www1.unece.org/cefact/platform/display/TBG/User+Guide+for+CCTS+3+and+NDR+3, 2010, Last revisited: Feb 2013.

[34] UN/CEFACT. *Techniques and Methodologies group*, http://www.untmg.org/working-groups/, Last revisited: Feb 2013.

[35] UN/CEFACT. *UN/CEFACT's Core Component Library (UN/CCL)*, http://www.unece.org/cefact/codesfortrade/unccl/CCL_index.htm, 2009, Last revisited: Feb 2013.

[36] M. Staron , L.Kuznaiaz. *Properties of Stereotypes from the Prespective of their Role in Designs*, Model Driven Engineering Languages and Systems, pp. 201-216, 2005.

[37] Liegl P. *Conceptual Business Document Modeling using UN/CEFACT 's Core Components*,in Proceedings of the Sixth Asia-Pacific Conference on Conceptual Modelling (APCCM2009), Wellington, New Zealand, Australian Computer Society, pp. 59-69, 2009.

[38] OASIS. *Registry Technical Specification 3.0*, `http://docs.oasis-open.org/regrep/v3.0/specs/regrep-rim-3.0-os.pdf`, 2005. Last Visit: Feb 2013.

[39] UN/CEFACT. *UN/CEFACT Registry Implementation Specification*, `http://www.uncefactforum.org/ICG/ICG_document_downloads.htm`, 2008, Last revisited: Feb 2013.

[40] UN/CEFACT. *XML Naming and Design Rules 3.0, ODP5, United Nations Center for Trade Facilitation and Electronis Business*, 2008.

[41] UN/CEFACT Core components data type catalogue 3.0, `http://www.untmg.org/ccdatatypecatalogue/spec_3_0`, 2009, Last revisited: Feb 2013.

[42] Lientz B.P., Rea K.P. *Dynamic E.Business Implementation Management*, pp. 3-23, 2000.

[43] Petersen A. *Patterns for e-business and the evolution of business to business on the Web*, IBM DeveloperToolbox Technical Magazine, 2001.

[44] Hofreiter B., Huemer C., Winiwarter W. *OCL-Constraints for UMM Business Collaborations*, E-commerce and web technologies: 5th International Conference, EC-Web 2004, Zaragoza, Spain, proceedings, Volume 5, August 31-September 3, 2004.

[45] Chesher M., Kaura R., Linton P. *Electronic Business & Commerce*, pp 340-341, 2003.

[46] Liegl P. *Conceptual Business Document Modeling using UN/CEFACT's Core Components,* in Proceedings of the Sixth Asia-Pacific Conference on Conceptual Modelling (APCCM2009), Wellington, New Zealand, Australian Computer Society, pp. 59-69, 2009.

[47] UN/CEFACT International Trade and Business Process Group (TBG14). *UN/CEFACT Common Business Process Catalog*, Version 0.95, 2003.

[48] UN/CEFACT Technologies and Methodologies Group (TMG). *UML Profile for UN/CEFACT's Modeling Methodology (UMM) Base Module*, Version 2.0

Technical Specification, http://www.untmg.org/umm/spec/base/2_0, May 2012, Last revisited: Feb 2013.

[49] UN/CEFACT Technologies and Methodologies Group (TMG). *Business Process Working Group (BPWG)*, http://www1.unece.org/cefact/platform/display/public/Business+Process+Working+Group+(BPWG), July 2012, Last revisited: Feb 2013.

[50] Miles R., Hamilton K. *Learning UML 2.0 - A Pragmatic Introduction to UML*, 2006.

[51] The Internet Engineering Task Force (IETF). The Internet Engineering Task Force RFC 3066, http://www.ietf.org/rfc/rfc3066.txt, 2001, Last revisited: Feb 2013.

[52] International Telecommunication Union. INTERNATIONAL STANDARD ISO/IEC 9834-8 ITU-T RECOMMENDATION X.667, http://www.itu.int/rec/T-REC-X.667-200409-S/en, 2004, Last revisited: Feb 2013.

[53] Bicevskis J., Cerina-Berzina J., Karnitis G., Lace L., Medvedis I., Nesterovs S. *Practitioners View on Domain Specific Business Process Modeling*, In: Frontiers in Artificial Intelligence and Applications, Vol. 224, IOS Press, 2011, pp. 169-182.

[54] Eriksson H., Penker M. *Business Modelling with UML*, John Wiley & Sons, New York, NY, ISBN 0-471-29551-5, 2000.

[55] Cortes-Cornax M., Dupuy-Chessa S., Rieu D. *Choreographies in BPMN 2.0: New Challenges and Open Questions*, 2012.

[56] Kopp O., Leymann F., Wagner S. *Modeling Choreographies: BPMN 2.0 versus BPEL-based Approaches*, 2011.

[57] Cortes-Cornax, M., Dupuy-Chessa, S., Rieu, D., Dumas, M.: *Evaluating choreographies in bpmn 2.0 using an extended quality framework. Business Process Model and Notation*, 103–117, Springer 2011.

[58] Decker G., Kopp O., Leymann F., and Weske M.. *Interacting services: from specification to execution.* Data & Knowledge Engineering, 68(10):946–972, April 2009.

[59] Liegl P., Huemer C. *State-of-the-art in business document standards*, 2010.

[60] OASIS Universal Business Language TC, https://www.oasis-open.org/committees/ubl/faq.php, Last revisited: Feb 2013.

[61] Nahid Jilovec. *Edi, Uccnet & Rfid: Synchronizing The Supply Chain*, 2004.

[62] UN/CEFACT Technologies and Methodologies Group (TMG). *UML Profile for UN/CEFACT's Modeling Methodology (UMM) Base Module*, Version 2.0 Technical Specification,`http://www.unece.org/cefact/umm/umm_index.html`, April 2011.

[63] C. Huemer, T. Motal, R. Schuster, H. Werthner. *From Economic Drivers to B2B Process Models: a Mapping from REA to UMM* In:Business Information Systems 13th International Conference, BIS 2010, Berlin, Germany, May 3-5, 2010. Proceedings, Springer Berlin Heidelberg, (2010), ISBN: 978-3-642-12813-4; S. 119 - 132.

[64] UN/CEFACT. *REA Specification Module for UN/CEFACT's Modeling Methodology(UMM)*, Public Draft V1.0, 2008.

[65] Zapletal M. *A UML-based Methodology for Model-Driven B2B Integration: From Business Values over Business Processes to Deployment Artifacts*, 2009.

[66] Zapletal M., Liegl P., Schuster R., *A UML Profile and Tool Support for UN/CEFACT's Modeling Methodology*, 2006.

[67] Gordijn J., Akkermans H. *Value based requirements engineering: Exploring innovative e-commerce idea*. Requirements Engineering Journal, 8(2):114–134, 2003.

[68] Geerts G., McCarthy W. E. *The Ontological Foundation of REA Enterprise Information Systems*. Technical report, Michigan State University, 2000.

[69] Osterwalder A., Pigneur Y. *An e-Business Model Ontology for Modeling e-Business*. In 15th Bled Electronic Commerce Conference, 2002.

[70] Pankaj R. Telang, Munindar P. Singh. *Comma: A Commitment-Based Business Modeling Methodology and its Empirical Evaluation*, 2012.

[71] Prasad M. Jayaweera, Ranjith Senaratne. *Mobile Service Portal for Rural Fisher Community Development*, 2011.

[72] Knittl S., Schaaf T., Saverchenko I. *Change Management in e-Infrastructures to Support Service Level Agreements*, 2012.

[73] Knittl S., Brenner M. *Towards a Configuration Management System for Hybrid Cloud Deployments*, 2011.

[74] UN/CEFACT. UMLProfileforCoreComponentsTechnicalSpecification3.0, Implementation Verification Step 6, 2011, http://www1.unece.org/cefact/platform/display/public/Core+Components+Working+Group+(CCWG), Last revisited: Feb 2013.

[75] Liegl P. *Business Documents for Inter-Organizational Business Processes*, 2010.